Techniques to Accelerate Boundary Element Contributions in Elasticity

Derek Scales

A Thesis presented for the degree of
Doctor of Philosophy

Advanced Mechanics
School of Engineering
University of Durham
England

March 2007

Dedicated to

Pat, Robert and Pete for their continued support throughout my education without them none of this would have been possible for me. I will be forever grateful.

To Mariko for helping me to see clearly when times were dark and ensuring that I never lost sight of the important things in life.

Ancora imparo

- *Michelangelo Buonarroti*

Techniques to Accelerate Boundary Element Contributions in Elasticity

Derek Scales

Submitted for the degree of Doctor of Philosophy
March 2007

Abstract

The problem of rapid re-analysis of small problems in elasticity is investigated. The aim is to enable updated stress contours to be displayed in real-time as a design geometry is dynamically modified. The focus of this work is small to medium sized problems; as a result it cannot be assumed that the solution phase dominates, and so the evaluation of boundary integrals is considered as well as the equation solution.

Two strategies are employed for acceleration of boundary element integrals: the use of Look-Up Tables (LUTs) containing precomputed integrals and the use of approximate analytical expressions derived from surface fits. These may be used in the matrix assembly and internal point calculations. LUTs are derived for both flat and circular arc elements for both the displacement and stress boundary integral equation. Details are provided on suitable LUT refinements and the approach is benchmarked against conventional Gauss-Legendre quadrature. The surface fit approach is presented as an alternative to LUTs that does not incur the considerable memory cost associated with LUTs. This approach has been limited to flat elements.

The equation solution is cast in a re-solution framework, in which we use a GMRES iterative solver. Convergence is greatly accelerated by using an approximate but complete LU preconditioner updated periodically using multi-threading. Consideration of the period of update is investigated with reference to the spread of

eigenvalues in the preconditioned system.

The resulting system achieves the aim of providing real time update of contours for small to medium size problems on a PC. This development is expected to allow a qualitative change in the way engineers might use computer aided engineering tools, in which design ideas may rapidly be assessed immediately as a change is made.

Declaration

The work in this thesis is based on research carried out at the Advanced Mechanics Group, School of Engineering, Durham, England. No part of this thesis has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Parts of this work have been submitted in the following:

Journals

D. J. Scales and J. Trevelyan. Techniques to accelerate BEM computation to provide virtual reality update of stress solutions. *Eng. Anal. with Boundary Elements.*

Conferences

J. Trevelyan and D. J. Scales and S. H. Spence. Interactive display of stress contours in real time. NAFEMS world congress, NAFEMS, May 2007.

J. Trevelyan and D. J. Scales. Rapid re-analysis in BEM elastostatic calculations. In C. A. Brebbia and J. T. Katsiukadelis, editors, *Boundary Elements and other Mesh Reduction Methods XXVIII*, pages 263–272. BEM/MRM 28, WIT Press, May 2006.

D. J. Scales and J. Trevelyan. Rapid re-analysis in 2D BEM elastostatic calculations. In K. Chen, editor, *Advances in Boundary Integral Methods: Proceedings of the 5th UK Conference on Boundary Integral Methods*, pages 153–162. UKBIM 5, University of Liverpool, September 2005.

J. Trevelyan, D. J. Scales, R. Morris, and G. E. Bird. Acceleration of boundary element computations in reanalysis of problems in elasticity. In Z. H. Yao, M. W. Yuan, and W. X. Zhong, editors, *Computational Mechanics: Abstract (Volume 2)*, page 44. WCCM VI in conjunction with APCOM '04, Tsinghua University Press and Springer-Verlag, September 2004.

Acknowledgements

There are lots of people I would like to thank for a huge variety of reasons.

It is impossible to overstate my appreciation and gratitude towards my PhD supervisor, Dr. Jon Trevelyan. Without Jon I feel that this work would not have been possible for me, his enthusiasm towards my work, the ideas and suggestions on how to deal with the problems that I have faced throughout my PhD have been inspirational. Without Jon I truly believe that this work would not have been possible.

Thank you to my examiners, Dr. Charles Augarde (internal) and Prof. Ferri Aliabadi (external), for managing to read the whole thesis so thoroughly, and for the feedback and comments that have improved this thesis. I hope that you found the experience useful and that I have provided an alternative perspective on the area.

Additionally, I would like to thank the staff at BAE SYSTEMS for sponsoring my PhD work. In particular I would like to thank Stuart Spence and the late Mike Henningsen for the feedback they gave as my work progressed. I also extend my thanks to Brian Oldfield and Chris Bingham for making my industrial placement at BAE Warton an enjoyable and productive experience.

I would also like to thank the many friends that I have made during my time at the University of Durham both as an undergraduate and more recently as a post-graduate. Without their continued support and belief in me I would have struggled

immensely. In particular I would like to thank Paul Jaquin for proof reading my thesis. I hope that my thesis was an enjoyable read and may help towards your own research.

I would like to thank Mariko. Your unfailing support and love throughout my PhD has kept me strong. When times seemed dark you always shone for me and showed me the way forward. You will always have a special place in my heart and I know that you will be a success in everything that you do.

I cannot end without thanking my family. The constant support offered by my parents, Pat and Robert, and my brother, Pete, has been unfailing even when times have been difficult. They have provided the constant encouragement and love that I have relied on throughout my time at University.

Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

α	Angle in 3D case
$\bar{L}$	3D element scale parameter
β	Angle in 3D case
$\boldsymbol{\epsilon}_0$	Initial strain vector
Δ	Dirac delta function
δ	Kronecker delta
$\Delta\mathbf{A}$	Modification matrix
$\Delta\mathbf{b}$	Modification vector
$\Delta\mathbf{x}$	Modification vector
η	Rotated coordinate system
Γ	Problem boundary
γ	Angular orientation of 3D element
γ	Euler constant
κ	Collocation point

κ	Matrix condition number
λ	Eigenvalue
$\bar{\mathbf{H}}$	Hessenberg matrix
$\mathbf{\Phi_1}$	Modal displacements
$\mathbf{A}'$	Perturbed version of $\mathbf{A}$
$\mathbf{A}$	Matrix of known coefficients
$\mathbf{a}$	Vector of coefficients
$\mathbf{B}$	FEM Strain-displacement matrix
$\mathbf{B}$	Matrix of known coefficients
$\mathbf{b}$	Vector of known displacements and tractions
$\mathbf{D}$	Material property matrix
$\mathbf{f}$	Vector of nodal forces
$\mathbf{G}$	Given's rotation matrix
$\mathbf{G}$	Influence matrix
$\mathbf{H}$	Influence matrix
$\mathbf{I}$	Identity matrix
$\mathbf{K}$	Global stiffness matrix
$\mathbf{L}$	Lower triangular decomposition
$\mathbf{M}$	Preconditioning matrix
$\mathbf{N}$	Shape functions
$\mathbf{p_k}$	Search vector
$\mathbf{P}$	SBFEM nodal forces

$\mathbf{p}$	Vector of meshless basis functions
$\mathbf{t}$	Traction vector
$\mathbf{u_h}$	SBFEM analytical functions in ξ coordinate direction
$\mathbf{U}$	Upper triangular decomposition
$\mathbf{u}$	Displacement vector
$\mathbf{x}$	Vector of unknown tractions and displacements
$\mathbf{y}$	Vector of known tractions and displacements
$\mathcal{E}$	Error term
$\mathcal{K}$	Krylov subspace
μ	Shear modulus
ν	Poisson's ratio
Ω	Problem domain
Φ	Iterative solver minimisation function
ϕ	Angle subtended by the element relative to the x coordinate
ϕ	Meshless methods shape functions
ϕ_i	Incident wave
σ	Stress tensor
θ	Angle subtended by the imaginary line r_m
ε	Error norm
ξ	Local coordinate variable
ξ	Scaled boundary finite-element method radial coordinate
ζ	Rotated coordinate system

a	Initial value for geometric progression
b	Body force
c	Coefficient representing how the surface varies at the source point
C_{1-5}	Constants based on material properties
D_{kij}	Stess fundamental solution
E	Young's modulus
E^0	SBFEM coefficient matrix
E^1	SBFEM coefficient matrix
E^2	SBFEM coefficient matrix
G	Green's function
g	Influence coefficient
G_{ij}	Galerkin tensor
h	Finite difference method step size
h	Influence coefficient
h	Simpson's rule strip width
$h_{i,k}$	Hessenberg matrix component
$J(\xi)$	Jacobian of integration
J	Meshless methods minimisation function
k	Wavenumber
K_1	Potential kernel
K_2	Potential kernel
L	Element length

L	Number of internal nodes
L_1	Length of side 1 of 3D elèment
L_2	Length of side 2 of 3D element
N	Number of equations in analysis model
N	Shape functions
n	Normal direction
n	Number of basis functions
P	ILU sparsity pattern
R	Radius of a circular arc element
r	Iterative solver residual
R_m	Scaling parameter
r_m	Distance from source point to the mid-point of the field element
S	Scaled boundary finite-element method defining curve
s	Circumferential coordinate for scaled boundary finite-element method
s	Geometric progression value
S_{kij}	Stess fundamental solution
t	Traction
T_{ij}	Displacement fundamental solution
U	Finite difference solution
u	Displacement
U_{ij}	Displacement fundamental solution
w_i	Gauss quadrature weight

x Coordinate variable

x Field Point

x_0 SBFEM Scaling centre location

x_i Gauss quadrature abscissa

y Coordinate variable

y_0 SBFEM Scaling centre location

Z ILU fill-in parameter

CHAPTER 1

Introduction

The aim of this thesis is to present a novel technique for accelerating the boundary element method, in particular with respect to stress analysis problems. The work will consider extensions to other areas to which the boundary element method has been applied.

1.1 Overview

Numerical analysis is an essential tool in the engineer's toolbox. It allows the solution of complex structures that do not have an analytical solution. This process tends to be computationally expensive and as a result a large number of authors have investigated methods of accelerating these computations.

Typically acceleration techniques have been applied to large problems for which the solution time is measured in hours, thus even a small reduction in computational cost within a loop can propagate to a large overall saving in run-time. As a result small *everyday* problems that have been considered trivial have been ignored. However, it is in the early stages of design, when simplifications have been applied to models for ease of analysis, that defining decisions are usually made. Thus if a

large number of alternatives can be tested and examined then late, costly, changes to a design can be avoided. Accelerating the solution methodology such that contour plots of stress and displacements update as the geometry is perturbed allows a much higher degree of interaction between user and design. Therefore this thesis will consider small two-dimensional problems and attempt to accelerate the associated computations. As a result the techniques employed to accelerate the solution will differ from those applied to large scale problems in certain aspects.

The solution of large problems is dominated by the solution of large systems of linear equations (Marburg and Schneider, 2003) and therefore the implementation of an iterative solver is the typical first step. Iterative solvers can be accelerated by application of a preconditioner of some form. The effectiveness of a preconditioner is problem dependent and thus a number of authors have investigated preconditioners for certain individual types of problem. For small problems, such as those considered in this work, run-time is no longer dominated by the solution of the global matrices and therefore an effective acceleration strategy needs to consider other areas of computational cost.

1.2 Background

To allow the analysis of the overall problem it is necessary to split the problem into the component parts and improve the computational cost of each segment.

1.2.1 Boundary element method

Although the boundary element method (BEM) is considered to be a relatively *young* method for analysing problems the initial work was laid down in a number of part papers by Somigliana (1885a,b,c,d, 1886a,b,c). After this initial work the method was slowly developed, primarily by mathematicians, for analytical and hence almost trivial problems until the 1960s. At this point research in the BEM (as well as other numerical methods) accelerated due to the more common usage of computing power at research institutes. This led to a number of key publications including Jaswon (1963) and Symm (1963) who developed a method of discretisation for the integral

equations. Later work came from Rizzo (1967) and then Cruse and Rizzo (1968) who extended the method to allow the direct use of tractions and displacements initially in 2D and then later in 3D (Cruse, 1973).

Application of the BEM involves a number of stages. The first requirement is to divide the boundary of the object under analysis into *elements*. Early work by Jaswon and Symm (1977) used constant elements, however the use of quadratic iso-parametric elements as proposed by Lachat and Watson (1976) is commonly used in modern analysis codes. Figure 1.1 shows a sample problem divided into its constitutive elements with each red dot representing the end of an element.

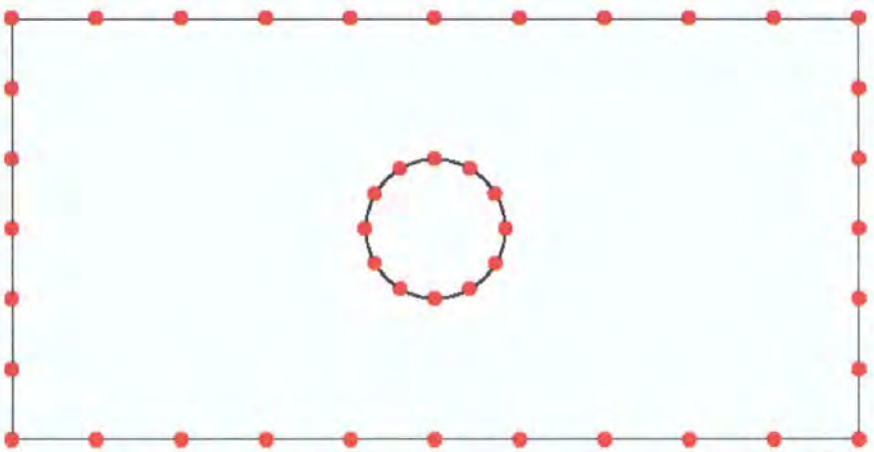


Figure 1.1: Sample problem discretized

To allow the solution of the boundary problem it is necessary to consider points on the boundary, called *collocation* or *source points*, and integrate around the remainder of the boundary. In performing this collocation it is possible to generate equations consisting of *influence coefficients* relating how forces applied to the source point will be transmitted to the boundary at specific locations, namely the elements (see figure 1.2).

Equation (1.1) shows the result of integrating at one of the source points around the remainder of the boundary

$$c_1 u_1 + h_{11} u_1 + h_{12} u_2 + \cdots + h_{1n} u_n = g_{11} t_1 + g_{12} t_2 + \cdots + g_{1n} t_n \qquad (1.1)$$

where h and g are the influence coefficients and c_1 represents the local geometry at the source point, these factors are all known. u and t are displacements and

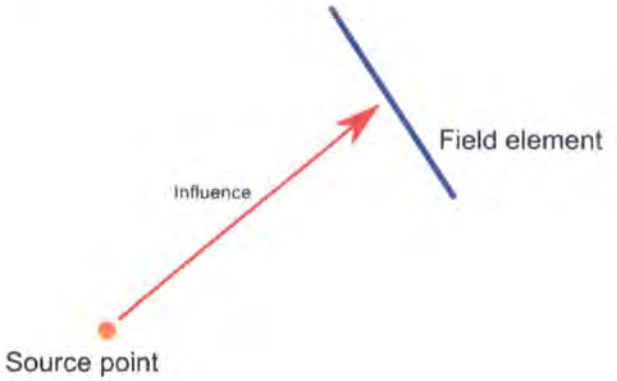


Figure 1.2: Source point - Field element pair

tractions respectively. By moving the source point around each of the nodes on the boundary in turn it is possible to generate a set of equations involving the tractions and displacements around the boundary. These can be combined into a matrix equation of the form

$$\mathbf{Hu} = \mathbf{Gt} \tag{1.2}$$

where $\mathbf{H}$ and $\mathbf{G}$ contain the respective influence coefficient matrices and $\mathbf{u}$ and $\mathbf{t}$ are vectors of displacements and tractions. Boundary conditions are now specified and rows and columns can be swapped

$$\mathbf{Ax} = \mathbf{By} \tag{1.3}$$

where $\mathbf{A}$ and $\mathbf{B}$ are mixtures of $\mathbf{H}$ and $\mathbf{G}$, $\mathbf{y}$ is a vector filled with known values of tractions and displacements, and $\mathbf{x}$ is a vector of unknown tractions and displacements. This can be further simplified by performing the matrix-vector multiplication $\mathbf{By} = \mathbf{b}$

$$\mathbf{Ax} = \mathbf{b} \tag{1.4}$$

This is a linear set of equations that can be solved in the user's preferred manner, to produce the remaining displacements and tractions from around the boundary.

To calculate displacements and stresses at points internal to the boundary it is necessary to collocate at the point of interest in a similar manner to the boundary problem, producing

$$u_{int} + h_{11}u_1 + h_{12}u_2 + \cdots + h_{1n}u_n = g_{11}t_1 + g_{12}t_2 + \cdots + g_{1n}t_n \tag{1.5}$$

where the only unknown is u_{int}. Thus the solution to the problem can be displayed to the user, and this can lead the user to decide upon developments within the geometric model. As the new perturbed model will be based on the previous model, which has already been solved, it is possible to re-use portions of the previous analysis to accelerate the solution.

Consideration of the overall solution strategy as a number of discrete steps has allowed each step to be considered in detail. Upon combining these steps together into the complete strategy the savings are combined.

The main strategy for problem solution within any analysis can be split into the following main stages,

- Define the problem geometry
- Convert the geometric problem to the numerical form
- Solve the numerical problem
- Convert the numerical solution to a suitable form for display
- Display the solution to the user

A stress analysis program developed in-house (Trevelyan, 2003) has been used as the basis for this research and therefore only the areas of main concern have been developed. Definition of problem geometry and the displaying of solutions is a relatively well developed area in the program.

The main areas that this thesis will consider are the formation of the global matrix problem and then the solution of this numerical problem. When considered in relation to the boundary element method the second of these stages can be further split into two sub-stages,

1. Solution of the boundary problem
2. Solution of the internal point solution

Each of these stages will be considered in turn.

1.2.2 Integration techniques

The rapid evaluation of integrals is a relatively un-researched area with most researchers preferring the brute force approach of calculating each stress or displacement *on-the-fly*.

Gauss-Legendre quadrature (e.g. Davis and Rabinowitz (1984)) is the most commonly used form of integration in numerical analysis due to the ability to accurately and relatively cheaply integrate functions. As a result of this popularity a large number of authors have investigated general Gauss quadrature rules aiming to improve the computational efficiency whilst maintaining a level of accuracy within the results obtained. Eberwien *et al.* (2005) considered the integration for source points internal to the problem domain. As the points are internal to the domain the integrals considered will not become singular. However, as the source points approach the boundary they will become near-singular. The order of the singularity depends on the particular boundary integral kernel being considered. Traditional BEM formulations include weakly and strongly singular integrals. More recent formulations include hyper-singular integrals. Eberwien *et al.* (2005) compared the proposed integration scheme with that of schemes previously proposed by Jun *et al.* (1985) and Bu and Davies (1995).

Telles (1987) proposed the use of a second or third degree polynomial transformation to remove the singular nature of the integral and allow the accurate calculation of the integral using a much reduced order of Gauss-Legendre integration. Telles applied the technique to integrals of the form $\mathcal{O}\left(\frac{1}{r}\right)$ and $\mathcal{O}\left(\frac{1}{r^2}\right)$. However, the technique can be transferred to integrals of logarithmic or higher order.

The second order transformation can be applied to integrals with logarithmic singularity at the extremity of the integral range. If the singularity is within the integral bounds then the integral can be split into two sections and the transformation applied to each part of the original integral. The third order transformation does not suffer from this restriction and as a result can be applied to more situations directly but at added computational cost.

Trevelyan and Wang (2001b) proposed the use of look-up tables (LUTs) to store precomputed values of influence coefficients for the displacement equations to allow

rapid compilation of the matrices. Trevelyan and Wang also implemented a scheme to reduce the size of the LUTs from a 3-parameter table to a 2-parameter table to describe all configurations of source point and field element in a 2D environment.

More recently work has concentrated on accurately solving near-singular and hyper-singular integrals using exact integration methods. In particular Zhang and Zhang (2004a,b) have produced formulae to allow the exact integration for stress analysis problems for linear and quadratic elements. The exact integration approach is an attractive idea as it allows the use of a single integration method for both the internal point problem and for the boundary problem. As the technique involves exact integrations the singularity that occurs during the integration phase is implicitly included in the resulting formulae. However, this ease of implementation comes at the additional computational cost that is required to compute the generic integral. For this reason a method based on exact integration is not suited to the real-time analysis that is intended in this research.

Tsamasphyros and Theotokoglou (2006) consider the integration of near singular integrals by modifying the Gauss quadrature scheme. The quadrature formula is modified by introducing an interpolatory formula and applying this to both regular and singular integrals where near-singularities are involved. The proposed scheme involves modification of the Gauss weights and thus the method is computationally more expensive than classical Gauss quadrature techniques. The advantage of the technique is the ability to solve the near-singular integrals accurately.

Takahashi *et al.* (2006) considered medium scale problems ($N \sim 10^5$) where the calculation time is dominated by the integration of the layer potentials in the Laplace and Helmholtz problem. They employed a specialist computer, MDGRAPE-2, to perform the integrals. The specialised design of the computer allowed the integration process to be optimised for the particular case concerned, and thus the computational cost of the integration phase was reduced dramatically. A standard PC was used to control MDGRAPE-2, collate the completed integrals and then perform the final iterative solution.

1.2.3 Matrix solution

Evaluation of the boundary integral equations around the problem boundary produces a linear system of equations. The solution of this matrix problem has been studied in some depth for a variety of different scenarios. There are two main types of solver,

- Direct
- Iterative

Direct methods such as Gaussian Elimination and Lower-Upper factorisation (Kreyszig, 1999; Pozrikidis, 1998; Stroud, 1995; Westlake, 1968) have not been specifically researched since invention as they guarantee to reach a solution within a specific number of operations $\mathcal{O}(N^3)$ where N is the number of equations.

Iterative solvers have been very popular, particularly for large systems of equations where the the use of a direct solver is not feasible either due to the long computation time or the high storage cost. Therefore the use of a method that can quickly come to an approximate solution is very desirable.

The conjugate gradient method (Axelsson, 1994; Kane, 1994; Pozrikidis, 1998; Prasad *et al.*, 1994) is an iterative solver created for the solution of symmetric systems commonly produced by methods such as the finite element method. The creation of conjugate search vectors allows the global minimum to be found for a particular problem. The derivation of the method (e.g. Kane, 1994) relies on the assumption of symmetry in the matrix $\mathbf{A}$ in the particular problem of $\mathbf{Ax} = \mathbf{b}$. To overcome this assumption of symmetry the normal equations ($\mathbf{AA^T}$) can be used. This makes the method more adaptable but with the disadvantage that the convergence rate is dependent on the square of the condition number of the original coefficient matrix.

In a similar vein is the use of a conjugate gradient squared method (Axelsson, 1994; Kane, 1994; Prasad *et al.*, 1994). This is advantageous because it will improve the performance of a standard conjugate gradient method. However, this is also a disadvantage, as if the system diverges under the conjugate gradient method, it will diverge rapidly using the conjugate gradient squared method.

Another extension of the conjugate gradient method is the stabilised bi-conjugate gradient (Bi-CGStab) method (van der Vorst, 1992). This allows for a smoother and more rapid convergence than standard conjugate gradient methods. The advantage of a bi-orthogonalisation method as opposed to a Lanczos based method is that the bi-orthogonal vectors can be used for un-symmetric systems. The methodology of Bi-CGStab is in a similar vein to that of the conjugate gradient squared method but instead of squaring the residual polynomials, they are updated with a linear factor. The introduction of this linear factor allows for information on the local behaviour to be introduced and introduces a smoothing effect on the convergence of Bi-CGStab.

An alternative to conjugate gradient based methods is the generalised minimum residual method, GMRES, proposed by Saad and Schultz (1986). This technique is efficient as it can be shown (Spencer, 2004) by application of the Cayley-Hamilton theorem that it will converge within N iterations with each iteration being of order N^2, due to a matrix-vector product in the iterative loop. Hence at worst it will be comparable with a direct solver. However, this relies on perfect precision in the numerical work. Additionally GMRES is suitable for dense un-symmetric systems of equations such as those generated through the BEM. Descriptions, including pseudocodes, of the main iterative methods is available in Axelsson (1994); Kane (1994) and Saad (1996).

Development with respect to solvers has continued aimed at improving rates of convergence of solvers under certain conditions. Preconditioning of the equations has been the most effective way of reducing iteration count and has been investigated from both a mathematical and an engineering point of view.

From a mathematical point of view the process has been to develop new and more efficient preconditioners for specific types of problems. Leung and Walker (1997) considered the use of diagonal preconditioning; they found that although the preconditioner was efficient it was possible that the basis vectors used in the iterative process of GMRES could suffer from a loss of orthogonality thus causing the method to fail. Leung and Walker found that the introduction of full re-orthogonalisation, proposed by Wang and Semlyen (1990), could delay the onset of divergence signifi-

cantly.

Valente and Pina (1998) have considered row-scaling as a form of preconditioning of the conjugate gradient method when applied to boundary element method matrices. They found that of the solvers implemented only bi-conjugate gradient, bi-CGStab and the conjugate gradient squared solvers converged. By application of preconditioning it was possible to accelerate the rate of convergence. The use of row-scaling as a preconditioner is simple to apply since the intention is to obtain a unit diagonal. Valente and Pina (1998) compared this technique with an incomplete factorisation based on a band-diagonal factorisation.

Chen (1998) considered the application of a variety of preconditioners to dense linear systems such as those produced by the boundary element method. The preconditioners proposed are all sparse preconditioners and therefore benefit from previously developed sparse techniques for matrix vector products and similar operations. Gilbert and Toledo (2000) also considered a variety of preconditioners and solvers for use in a a *black box* environment. Black box environments raise a number of issues as preconditioners and iterative solvers tend to be optimal for certain categories of problems, and so finding a good all round solver and preconditioner is an extremely challenging task. Gilbert and Toledo (2000) concluded that for a black box environment (with no *a priori* knowledge) it would be more practical to utilise a direct solver (such as the back slash operator in MatLab (The MathWorks Inc., 2006)) as this will be able to consistently solve a non-singular matrix problem although potentially not in the optimal solution time.

Application of these methods to engineering problems has also been investigated by a number of authors. Prasad *et al.* (1994) looked at a variety of preconditioned Krylov solvers for the solution of both stress and thermal problems. They studied the form of the matrix and consider how the matrix terms are affected by different geometric features. From this it is possible to see that, although the boundary element method tends to lead to a strong diagonal dominance within the matrix equations, it is not always possible to employ merely diagonal preconditioning. This is a result of geometric features leading to cross-diagonal relationships (diagonals in the opposite direction to the leading diagonal) within the matrix terms.

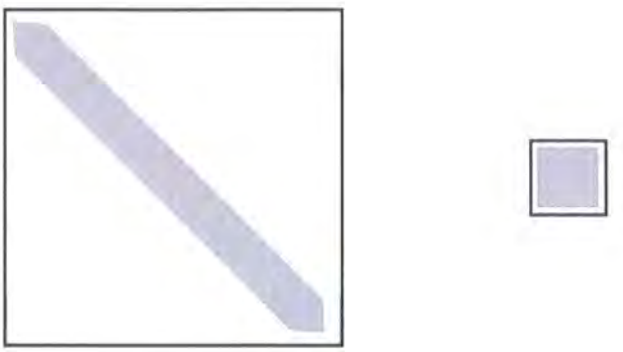
Merkel *et al.* (1998) considered the use of iterative solvers for large-scale three-dimensional industrial problems. From the study of more complex features it was found that the diagonal dominance that is commonly found in smaller, simpler problems vanishes and the distribution of eigenvalues is a good indicator for convergence properties for these types of problems. Clustering of the eigenvalues has been noted by a number of authors (e.g. Chen (1994); Greenbaum (1979)) to accelerate the convergence of iterative solvers and as such they are a good overall indicator of convergence rates. Merkel *et al.* (1998) found that for large problems it was important to order the equation system to reach convergence.

The use of sparse preconditioners for BEM systems approximating the three-dimensional Helmholtz problem are presented by Chen and Harris (2001). The preconditioners are based on mesh-neighbour preconditioners and as a result can be thought of as a variable width banded matrix type preconditioner. Chen and Harris (2001) found that the application of these preconditioners resulted in better clustering of the eigenvalues for the normal equation matrix.

For acoustic problems Marburg and Schneider (2003) and Schneider and Marburg (2003) presented findings on a number of problems using both a simple diagonal preconditioner and an incomplete LU factorisation type preconditioner; of these they found that the diagonal preconditioner showed no benefit, and in some cases degraded, the rate of convergence of the iterative solver. They did however find that an incomplete LU factorisation based method was extremely effective at reducing iteration count. Chen and Waubke (2004) also considered the use of preconditioners in acoustic problems but using the fast multipole method. Chen found that the use of an incomplete LU factorisation was an extremely effective method for iteration count reduction.

It should be noted that the reduction of iteration count is a useful technique to decrease the overall computational cost as the iterative phase is a relatively costly stage of the solution. However, if the reduction of iteration count results in the calculation of an expensive preconditioner, for example the use of the inverse matrix will reduce the iteration count to only one iteration but the calculation of the inverse is a computationally prohibitive.

The use of sparse matrix methods has been made very popular by a number of authors including Saad (1996). This is a result of their applicability to the relatively more mature finite element method, FEM. This applicability is generated by the FEM producing large symmetric but banded matrices compared with the BEM which produces small but densely filled matrices, figure 1.3.



(a) FEM (b) BEM

Figure 1.3: Comparison of matrices - Shaded area indicates non-zero matrix terms

Sparse matrix methods are extremely popular due to the relatively low computational cost associated with their calculation and implementation. They are also extremely effective for use with matrices that are already primarily sparse in nature, such as those generated through application of the FEM.

1.2.4 Reanalysis

Reanalysis is the technique of using information calculated in a previous analysis to reduce the computational cost in subsequent analyses. Typically in reanalysis problems an iterative solver such as a BiCG variant or GMRES (Trevelyan and Wang, 2001a) is implemented. Iterative solvers are efficient in the cases of reanalysis due to the small difference in the matrix equations. The small difference allows the previous solution to the matrix equation to be used as the initial solution to the new perturbed matrix equation. This technique can be applied to both large and small perturbations with significant acceleration in computation.

Kirsch and Toledano (1983) present a number of techniques for reanalysis for modifications to the structural geometry of problems. Current techniques are dis-

cussed and a scheme based on a simple iteration with scaling is presented and compared. The computational cost with respect to the accuracy of each technique is presented. They conclude that the discussed techniques can be divided into three main bands with respect to accuracy and computational cost.

1. Polynomial fitting.

2. Simple iteration with scaling.

3. Taylor series.

Thus, they conclude that for most cases a scheme based on simple iterations and scaling is suitable. Kane *et al.* (1990) also presented an iterative method reanalysis for BEM systems. The method can be considered as the expansion of the perturbed system.

$$\begin{aligned} \mathbf{A}\mathbf{x} &= \mathbf{b} \\ (\mathbf{A} + \Delta\mathbf{A})(\mathbf{x} + \Delta\mathbf{x}) &= (\mathbf{b} + \Delta\mathbf{b}) \\ \mathbf{A}\Delta\mathbf{x} &= (\Delta\mathbf{b} - \Delta\mathbf{A}\mathbf{x}) - \Delta\mathbf{A}\Delta\mathbf{x} \end{aligned} \tag{1.6}$$

The term in brackets is known entirely and as $\Delta\mathbf{x}$ is on both sides of the equation an iterative scheme can be implemented. Kirsch and Toledano (1983) proposed the scaling of $\mathbf{A}$ to improve the convergence properties and hence allow more substantial changes.

Leu (1999) presented an iterative scheme based on a reduction method for problems of shape optimisation. The solution vector is composed of a linear combination of basis vectors in a similar manner to conjugate gradient methods. This is advantageous as only a reduced basis set is required to achieve the desired accuracy.

Both of the methods presented by Kane *et al.* (1990) and Leu (1999) suffer from being based on a factorisation of the original matrix. As a result any subsequent reanalysis is linked back to the original problem and hence as multiple perturbations are applied the solution technique can degrade. It is possible to update the factorisation in an evolutionary sense. However, this negates the need for such an iterative scheme.

Trevelyan and Wang (2001a) presented initial work with reanalysis associated with the boundary element method. The original analysis was organised such that appropriate rows and columns could easily be updated depending on the relevant perturbations made to the analysis model. This method is extremely effective as a means of reducing computational cost whilst being flexible to allow relatively large changes to be made to the model's geometry. Addition of features, and hence elements, to the model was achieved by the addition of rows and columns to the bottom of the matrix equations. This approach has been adopted in the Concept Analyst software (Trevelyan, 2003). Cervera (2003) used this technique to accelerate the analysis phase of an evolutionary stress optimisation (ESO) code.

An alternative method for accelerating the solution of the matrix equations, proposed by Bae *et al.* (2006), employs the successive matrix inversion (SMI) method to allow a quick updating of the matrix inverse. This technique considers that the difference between successive analysis will be small and therefore the matrix $\mathbf{A}'$ can be approximated by

$$\mathbf{A}' = \mathbf{A} + \Delta\mathbf{A}$$

where $\Delta\mathbf{A}$ is the modification matrix. As the modification is relatively small the difference in the inverses of $\mathbf{A}$ and $\mathbf{A}'$ will also be small, thus, Bae *et al.* (2006) use the SMI to accelerate the updating of the matrix inverse and as such the matrix solution. The SMI method can be employed with an iterative solver to improve the numerical conditioning of the system and improve the rate of convergence of the iterative solver.

The use of reanalysis for early design development has also been implemented by Terdalkar (2003); Terdalkar and Rencis (2006) using the FEM and the commercial package ANSYS®. This work improves the interactivity between engineer and design from previous solutions by allowing the adjustment of node positions within the finite element mesh before commencing a reanalysis. Previously this would have required a second full analysis after the adjustments had been implemented. The work is, however, linked to a very precise level of interaction where the user must control nodal positions as opposed to a more global view of the geometry. As such it is the view of the author that, although the work by Terdalkar and Rencis (2006)

is a big step in the right direction of greater interactivity, the work presented in this thesis moves to another level in interactivity, with contour plots being updated as the user interacts with the geometry.

Margetts *et al.* (2005) applied concepts of reanalysis together with parallel processing to create an interactive finite element analysis package. To allow the distinction between levels of analysis Margetts *et al.* (2005) defined three styles of analysis

- Steering
- Interactive
- Real-time

Steering involves computations that are significant in overall length and on account of this results are extracted as the analysis runs to consider if the solution is converging in the correct manner. The results extracted can be used to control, *steer*, the problem to a good solution. Due to the length of analysis it is also necessary to have completed extensive background work to allow an educated mesh to be created or to rely on the user's knowledge base and ability to produce a suitable design and mesh capable of meeting the design specification.

Real-time solutions of problems can typically be computed within the time it takes to refresh the visualisation environment, for example computations of order 10^{-2} s. Interactive simulations lie in-between these two extreme cases and are considered periods of computation for which the user will be happy to sit and wait for the analysis to complete. Interactive and real-time analysis promotes decision making based on the current analysis results. As a result of this, the user can quickly adapt a model and converge to a finalised product design quicker than relying solely on the users knowledge base and experience to generate a good design for the provided specification.

Margetts *et al.* (2005) found that to achieve the necessary speed of computation they required the use of large parallel computers over which to spread the computations. Thus, this technique is limited to organisations that have such facilities.

1.2.5 General implementation

A number of authors have presented object oriented methods applied to the solution of numerical analysis problems. The advantages of object oriented programming (OOP) are the ability to apply techniques such as

- Encapsulation
- Inheritance
- Polymorphism

Each of these ideas allows the rapid development of software, whilst maintaining ease of upgradability within the code. This is a result of the reduction in code replication within the same code base.

Mackie (1998) proposed the use of OOP for the implementation of a fully interactive finite element system aimed at the solution of smaller, computationally less intensive problems. In particular Mackie (1998) discusses the ability of OOP methods to allow for the sub-structuring of the finite element problem such that it can be divided between multiple threads of execution. Use of multiple threads allows the idle time of the processor to be minimised on single core machines on account of the ability of the operating system to give time-slices on the processor to any tasks that are waiting. As soon as a sub-structure has been defined the analysis on that sub-structure can start whilst the main thread of execution continues to define other substructures. As a result of this execution style it is necessary to ensure that threads terminate correctly and within good time otherwise one of three scenarios can occur,

- the system could crash,
- the user could be kept waiting for a thread to terminate, or
- incorrect values could be used in the calculation resulting in an incorrect answer for a particular geometry.

The use of multi-threading is beneficial for tasks with which multiple items can be computed simultaneously. If multiple processor cores are available then this will

allow the simultaneous computation, for single core machines tasks will be allocated time slots on the processor until computed.

Marczak (2004) shows how to implement OOP methodology for boundary integral equation methods but stresses that the auxiliary classes derived are equally applicable to other types of problems such as the finite element method. A further benefit of generic auxiliary classes is that they can be extensively tested in the production stage and test functions produced to ensure that the functions perform as designed, then they can be *packaged* and employed in larger projects. If a class then requires further development it can either be used as a base class for a new class; which will inherit all of the associated features of the original class, or if the external facing functions will not be altered (for example improvement of error checking which can be kept entirely within the class) then the class can be modified directly. Marczak (2004, 2006) also notes the advantage of using a template based system for functions such as integration as the system can be implemented such that a variety of objects can be passed into the function.

The use of OO techniques has been applied for problems of reanalysis (Trevelyan and Wang, 2001a) where a model is perturbed such that the new model is close to the original model. Use of OOP allows model data to be related to higher order abstractions. For example, nodes can be linked to elements which in turn are linked to geometric shapes. As a consequence, only data that are changed need to be updated within the new model. These techniques have been implemented within the Concept Analyst software (Trevelyan, 2003) and form the foundation for the implementation of the work presented in this thesis.

1.3 Outline of the thesis

In chapter 2 a number of numerical methods will be introduced, indicating the basic methodology behind each of them, leading to reasons why the boundary element method has been chosen as the numerical method of choice for this research. Chapter 3 will present the boundary element method in detail introducing notation relevant for the thesis. Chapters 4 and 5 will introduce additional material required for the

main portion of the thesis, introducing numerical integration techniques and solution methods for matrix equations respectively.

Chapters 6 and 7 will introduce material that has been developed through the research and present the relevant methodologies. Results will be presented in chapter 8 for the newly proposed techniques comparing them with current standard methodologies for both computational speed-up and numerical accuracy. Chapter 9 will consider extension of the proposed techniques to alternative applications outside of elasticity.

1.4 Directions for current work

A large amount of previous work has been presented in section 1.2 regarding the acceleration of the solution phase in particular. However, this work has tended to be aimed at either large problems or sparse problems; such as those generated by the finite element method. The main aim of this thesis is to present the techniques that have been developed to enable a real-time analysis of two dimensional stress analysis problems.

For small problems which can easily be stored directly in memory the solution process can be divided into three roughly equal portions:

- Calculation of the terms in the matrix equations
- Solution of the matrix equations
- Solution of the problem at points internal to the problem domain to allow accurate contour plots to be generated

These portions can be further divided into two main categories:

1. Integration dependent
2. Solver dependent

As such these two tasks will be the main focus of the work presented in this thesis. In chapter 6 techniques for accelerating the computation of boundary integrals using

look-up tables will be presented. Chapter 7 will concentrate on the acceleration of the solution phase by utilising a preconditioned iterative solver.

CHAPTER 2

Numerical Methods in Stress Analysis

Numerical methods are an important part of the engineer's toolkit as they allow the solution of complicated problems which otherwise could not be solved, i.e. there is no analytical solution. To allow for the variety of different problems there are a multitude of different methods that can be employed. Each of these methods have their advantages and disadvantages making them more or less suitable for certain types of problems.

Examples of the types of problems that can be solved using numerical methods for any domain include Poisson's equation (equation (2.1)), Laplace's equation (equation (2.2)), the Helmholtz equation (equation (2.3)) and Navier's equations (equation (2.4)). Without the application of approximate numerical methods these problems could only be solved analytically for very simple cases.

$$\nabla^2 \phi = f \tag{2.1}$$

$$\nabla^2 \phi = 0 \tag{2.2}$$

$$\nabla^2 \psi + k^2 \psi = 0 \tag{2.3}$$

$$\frac{\partial^2 u_i}{\partial x_j \partial x_j} + \left(\frac{1}{1-2\nu}\right) \frac{\partial^2 u_j}{\partial x_i \partial x_j} = -\frac{f_i}{\mu} \tag{2.4}$$

where ϕ, f and ψ are functions defined within Euclidean space. k is the wavenumber, u are displacements, ν is Poisson's ratio, μ is the shear modulus and f_i are the components of the body force vector. In this chapter a number of methods will be presented and the associated advantages and disadvantages discussed. The methods considered are

- Finite Difference Method (FDM)
- Finite Element Method (FEM)
- Boundary Element Method (BEM)
- Scaled Boundary Finite Element Method (SBFEM)
- Meshless Methods

2.1 Finite difference method

The finite difference method involves the devolution of the differential equations involved to simpler difference equations that can be solved for a grid of points through the domain of interest. This conversion is achieved by considering the derivative of a function, $f(x)$

$$\lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{2.5}$$

where h is the step size. In the finite difference method h has a finite value rather than an infinitesimal value.

The value for the derivative at x can be approximated in three ways,

- Backward difference

$$U'(x) \approx \frac{1}{h} \{U(x) - U(x-h)\} \tag{2.6}$$

- Forward difference

$$U'(x) \approx \frac{1}{h} \{U(x+h) - U(x)\} \tag{2.7}$$

- Central difference

$$U'(x) \approx \frac{1}{h}\{U(x+h) - U(x-h)\} \tag{2.8}$$

Backward (equation (2.6)) and forward (equation (2.7)) difference methods have an error $\mathcal{O}(h)$ whereas the central difference method (equation (2.8)) has an error of $\mathcal{O}(h^2)$.

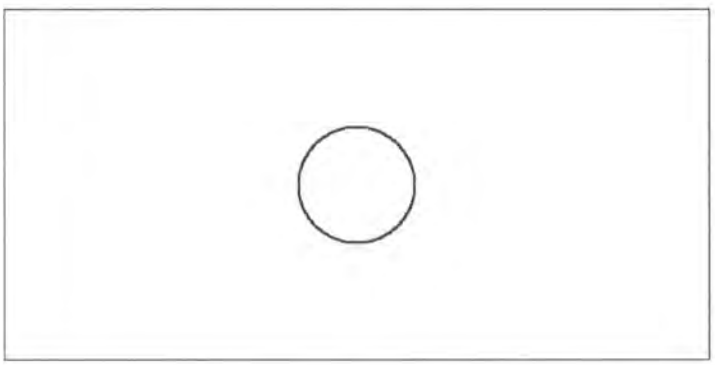


Figure 2.1: Sample problem

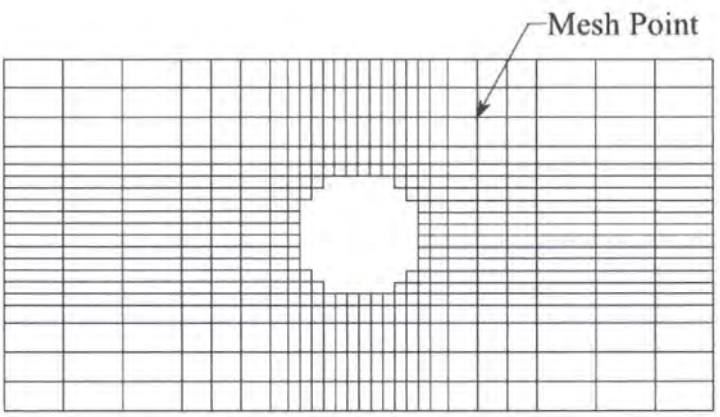


Figure 2.2: Typical finite difference mesh for the sample problem

The use of finite-differences is popular in computational fluid dynamics (CFD) (Fletcher, 1988) as it is a quick and easy method to apply a time-stepping formula to, as required for efficient analysis under CFD. A disadvantage of the finite difference method is the requirement of a structured mesh for the analysis (figure 2.2) and as a result meshing of complicated structures can be computationally difficult as the mesh needs to be constructed entirely from quadrilateral elements.

2.2 Finite element method

The finite element method (FEM) is a method for solving a large range of problems as varied as simple linear static calculations through to highly non-linear impact analysis. Due to the large range of problems that can be solved there is also a large amount of software available specialising in each of these areas.

The finite element method works by sub-dividing the domain of interest into smaller elements over which the constitutive equations can be applied and solved. As the domain is subdivided throughout the volume for a three-dimensional problem or across the surface for a two-dimensional problem this allows the method to approximate the non-linear behaviour that can occur. Additionally this sub-division across the domain means that the matrix equations generated will be large; due to the high number of elements involved. Figure 2.3 shows a typical finite element mesh for the sample problem shown in figure 2.1.

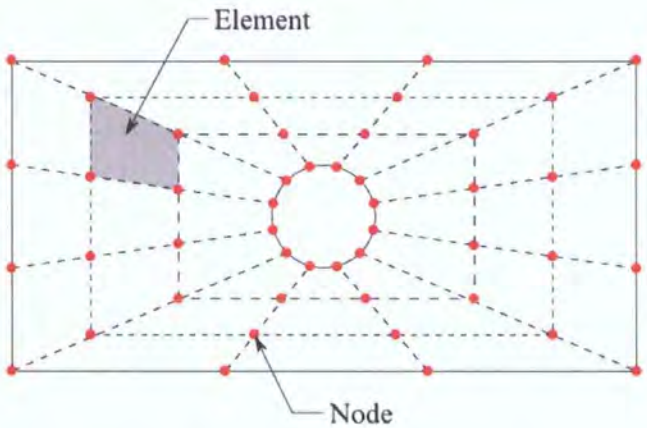


Figure 2.3: Typical finite element mesh for the sample problem

The typical form of the global matrix equations generated through the FEM are (Zienkiewicz and Taylor, 1989)

$$\mathbf{Ku} = \mathbf{f} \tag{2.9}$$

where $\mathbf{K}$ is the global stiffness matrix, $\mathbf{u}$ is the global displacement vector and $\mathbf{f}$ is the vector of nodal forces. $\mathbf{K}$ is obtained by assembling the individual stiffness matrix for each element, $\mathbf{K_e}$ (Zienkiewicz and Taylor, 1989)

$$\mathbf{K_e} = \int_{\Omega} \mathbf{B}^T \mathbf{D} \mathbf{B} d\Omega \tag{2.10}$$

where **B** is the strain-displacement matrix and **D** is the material property matrix (or constitutive matrix). **B** is dependent on the type of elements employed in the modelling procedure. Assembly of the individual stiffness matrices for each element causes connectivity to be achieved throughout the mesh. As a result the matrix equations that are produced, although large, are of a highly banded nature and therefore specialist solvers and preconditioners can be applied to ensure a rapid solution. Additionally this banded nature can be exploited to reduce the storage requirements of matrices by employing techniques such as compressed row storage or compressed column storage (Duff *et al.*, 1986).

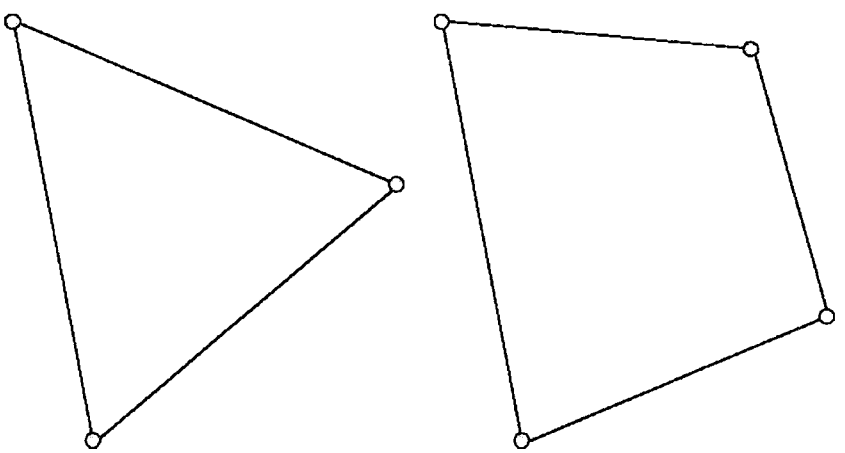
Displacements are converted to strains and then stresses are extracted for display

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{u} \tag{2.11}$$

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\varepsilon} + \boldsymbol{\sigma}_0 \tag{2.12}$$

where $\boldsymbol{\sigma}_0 = -\mathbf{D}\boldsymbol{\varepsilon}_0$ and $\boldsymbol{\varepsilon}_0$ is the vector of initial strains.

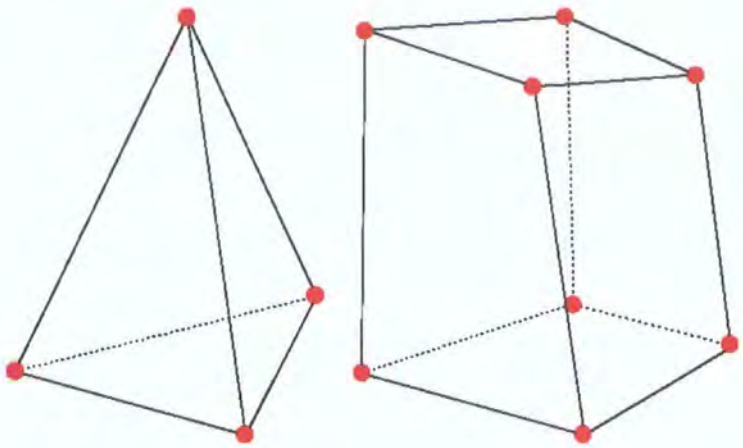
The most common elements employed in the finite element method are triangular or quadrilateral for 2-dimensional problems (figure 2.4) and tetrahedral or hexahedral for 3-dimensional problems (figure 2.5). The most popular element is



(a) Triangular element (b) Quadrilateral element

Figure 2.4: Typical 2-dimensional finite elements

the triangular (or tetrahedron for 3-dimensional problems) as it can be easily conformed to a wide variety of geometries without excessively distorting the element



(a) Tetrahedral element (b) Hexahedral element

Figure 2.5: Typical 3-dimensional finite elements

(figure 2.6 shows a quadrilateral element being greatly distorted). Distortion in an element is undesirable as it can cause problems from a numerical point of view. The disadvantage of using triangular elements is that typically a much greater number of elements is required to reach an appropriate degree of error in the numerical solution. This increase in element count will naturally increase the problem size and computational run-time.

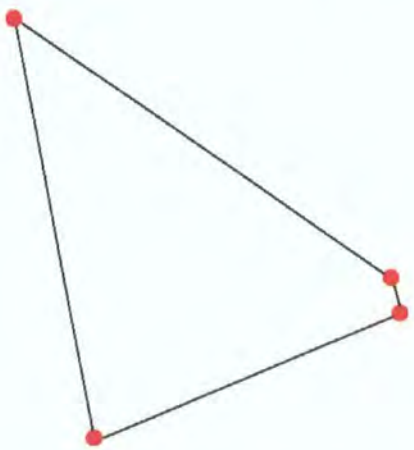


Figure 2.6: Quadrilateral element being distorted

For each of these types of element they can be further categorised into the number of nodes per element, so that elements of different polynomial order can be defined.

Although the finite element method is an adaptable method, and as such very

popular, it requires the domain to be subdivided into small individual elements (figure 2.3). When a problem is being remeshed after a reanalysis there is therefore a significant computational cost. If the perturbation is small; as would be the case in a real-time analysis, then this problem can be minimised by only remeshing areas of the domain that are affected. In 3-dimensional problems this is a much more complex task.

The FEM is a widely used technique within the numerical methods field due to the ease of application to a wide variety of problems. This is a result of the FEM being applied to the solution of almost any differential equation by merely casting the problem in the weak form and applying a weighted residual method (Ottosen and Petersson, 1992).

2.3 Boundary element method

This section presents a greatly condensed overview of the boundary element method (BEM) formulation in elastostatics. The material is covered in more detail in chapter 3.

The BEM is formed by taking the constitutive equations for a particular problem and applying methods to reduce the dimension of the integrals by one; reducing volume integrals to surface integrals and thus reducing the complexity of the mesh by one dimension. Figure 2.7 shows a typical boundary element mesh for the sample problem in figure 2.1.

As the number of elements has been reduced the overall size of the problem has been decreased. However, as the method now relates all of the problem's elements to every other element in the problem, the matrices formed will be fully populated.

The BEM uses the reciprocal nature, in a virtual work sense, of two load cases to calculate displacements and tractions associated with the problem. Betti's reciprocal theorem states

$$\int_\Gamma t_i u_i^* \mathrm{d}\Gamma + \int_\Omega b_i u_i^* \mathrm{d}\Omega = \int_\Gamma t_i^* u_i \mathrm{d}\Gamma + \int_\Omega b_i^* u_i \mathrm{d}\Omega \tag{2.13}$$

where Ω is the problem domain, Γ is the boundary of the problem domain, u is a displacement vector, t is a traction vector and b is a body force vector. To allow

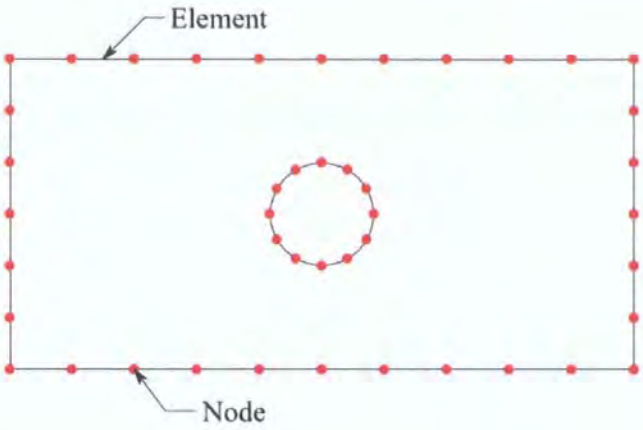


Figure 2.7: Typical boundary element mesh for the sample problem

the solution of this problem it is necessary to define a real and a fictitious (denoted by $*$ in equation (2.13)) load case. Thus if one load case is known completely then we can quickly establish the second more complicated load case. For the BEM we choose a fictitious load case to be the Dirac delta function. The Dirac delta function has a number of properties that make it useful as the fictitious load case. It

- reduces one of the volume integrals to a trivial form, and
- has analytical solutions for use in the remaining integrals.

As a result in the absence of body forces Betti's reciprocal theorem can be reduced to the boundary integral equation.

$$c_{ij}u_i + \int_\Gamma T_{ij}u_j \mathrm{d}\Gamma = \int_\Gamma U_{ij}t_j \mathrm{d}\Gamma \tag{2.14}$$

where c_{ij} is a coefficient dependent on the boundary geometry. Equation (2.14) can now be integrated around the boundary by placing the *collocation* point at each node in turn and integrating across all of the elements in the model. This produces a set of equations (one for each collocation point) which can be combined into the linear set.

$$\mathbf{Hu} = \mathbf{Gt} \tag{2.15}$$

Application of boundary conditions will reduce the number of unknowns such that there are N unknowns and N equations in the set which can be rearranged into the

well known linear form.

$$\mathbf{Ax} = \mathbf{b} \tag{2.16}$$

Equation (2.16) can be solved for the unknown displacements and tractions around the boundary. To calculate the displacements internal to the problem domain equation (2.14) is applied at the point of interest. Stresses can be extracted by use of the stress form of the boundary integral equation,

$$\sigma_{ij} + \int_{\Gamma} S_{kij} u_k \mathrm{d}\Gamma = \int_{\Gamma} D_{kij} t_k \mathrm{d}\Gamma \tag{2.17}$$

where S and D are third order tensors.

As only the boundary needs to be discretized then meshing of problems is a relatively simple task to achieve, for the two-dimensional problem it is merely a case of dividing the boundary line into elements. This simplicity is easily extended into three-dimensions in which the boundary is represented by a surface area that is discretised into surface elements.

2.4 Scaled boundary finite element methods

The scaled boundary finite element method (SBFEM) is a semi-analytical method that aims to take the best of both worlds from the FEM and BEM. In this method a model is converted to a coordinate system in which an analytical solution can be fitted in one of the coordinate directions. This coordinate system is called the scaled boundary coordinate system.

The scaled boundary coordinate system is formed by defining a single defining curve S. This can then be scaled throughout the domain via a scaling centre. The defining curve can be open or closed but must be smooth and C^0 continuous. A circumferential coordinate s is defined around the defining curve. A second coordinate ξ is radial in direction from the scaling centre and takes the value $\xi = 1$ on the defining curve. As such any point in the domain can be described in terms of (ξ, s).

There are three main cases that can be considered. The internal boundary ξ_i is taken to zero and as such the domain is bounded ($0 \leq \xi \leq 1$) and contains the scaling centre (figure 2.9(a)) or the external boundary ξ_e tends to infinity and as

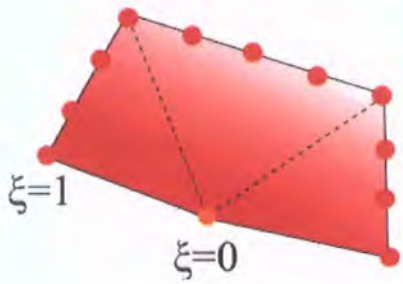


Figure 2.8: Scaled boundary coordinate system

such the domain is bounded by $(1 \leq \xi \leq \infty)$ and does not contain the scaling centre (figure 2.9(b)). The third case involves a combination of the previous two cases and has both an upper and lower bound (figure 2.9(c), Deeks (2002)).

The defining curve can be specified by $(x_0 + x_s(s), y_0 + y_s(s))$ and so we can relate the scaled boundary finite element method coordinates to the Cartesian co-ordinate set by the following

$$x = x_0 + \xi x_s(s)$$

$$y = y_0 + \xi y_s(s)$$

where the scaling centre is located at (x_0, y_0).

The scaled boundary finite element method aims to find an approximate solution of the form,

$$\mathbf{u}(\xi, s) = \sum_{i=1}^{n} N_i(s)\, u_i(\xi) = \mathbf{N}(s)\, \mathbf{u}(\xi) \tag{2.18}$$

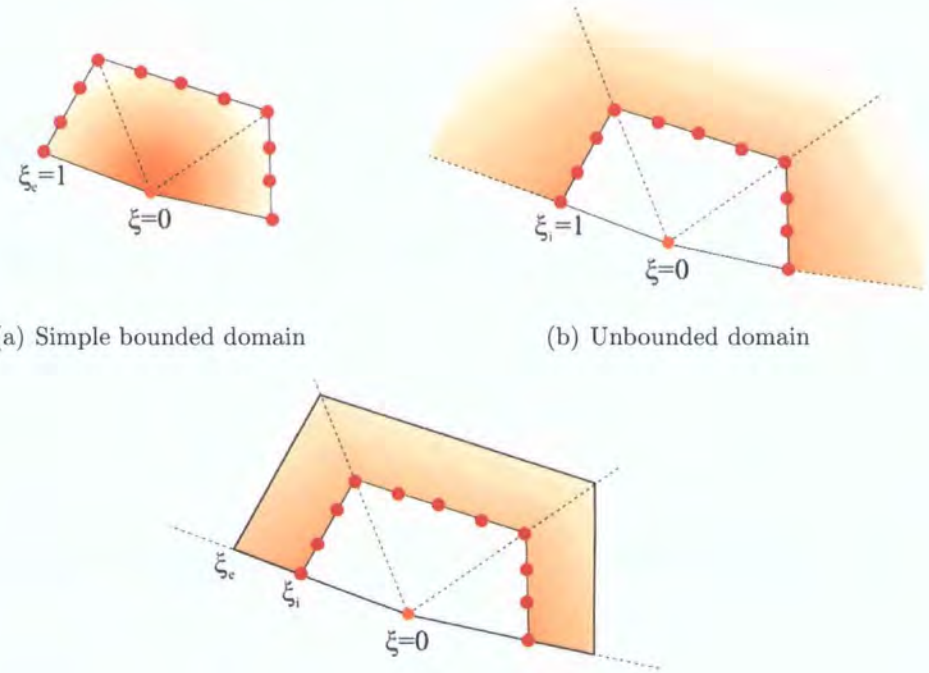
where $\mathbf{N}(s)$ represents the shape functions in the circumferential direction. $\mathbf{u}(\xi)$ represents the analytical displacements along the node lines as they extend from the scaling centre through the nodes on the boundary.

A finite element approximation is employed in the s coordinate direction.

Application of virtual work methods leads to the scaled boundary finite-element equation in displacement.

$$\mathbf{P} = \mathbf{E^0}\mathbf{u_{h,\xi}} + \mathbf{E^1}^T\mathbf{u_h} \tag{2.19}$$

$$\mathbf{E^0}\xi^2\mathbf{u_h}(\xi)_{,\xi\xi} + \left[\mathbf{E^0} + \mathbf{E^1}^T - \mathbf{E^1}\right]\xi\mathbf{u_h}(\xi)_{,\xi} - \mathbf{E^2}\mathbf{u_h}(\xi) = 0 \tag{2.20}$$



(a) Simple bounded domain

(b) Unbounded domain

(c) Upper and lower bounded domain

Figure 2.9: Different modelling strategies for the SBFEM

where $\mathbf{u_h}$ is a set of n analytical functions in ξ. $\mathbf{E^0}$, $\mathbf{E^1}$ and $\mathbf{E^2}$ are coefficient matrices.

This produces a weakened form of the governing equations in the circumferential direction but keeps a strong form in the radial direction.

Equations (2.19) and (2.20) can be considered to be a set of Euler-Cauchy differential equations, consideration of which yields the quadratic eigenproblem,

$$\left[\lambda^2\mathbf{E^0} - \lambda\left[\mathbf{E^1}^T - \mathbf{E^1}\right] - \mathbf{E^2}\right]\boldsymbol{\phi} = 0 \tag{2.21}$$

where $\boldsymbol{\phi}$ is a vector containing the modal displacements at the boundary nodes. This can be converted to a standard eigenproblem at the expense of doubling the size of the system (Deeks and Wolf, 2002). The solution of this standard eigenproblem yields $2n$ modes. For a bounded domain it is only necessary to consider the modes with non-positive real components of λ lead to finite displacements at the scaling centre. This subset of modal displacements is designated by $\mathbf{\Phi_1}$. For a particular set of boundary displacements $\mathbf{u_h}$ and nodal forces $\mathbf{P}$ we can state,

$$\mathbf{P} = \mathbf{Q_1}\mathbf{\Phi_1}^{-1}\mathbf{u_h} \tag{2.22}$$

thus, the stiffness matrix is given by,

$$\mathbf{K} = \mathbf{Q_1}{\mathbf{\Phi}_1}^{-1} \tag{2.23}$$

and equilibrium is reduced to

$$\mathbf{K}\mathbf{u_h} - \mathbf{P} = 0 \tag{2.24}$$

Application of boundary conditions reduces this problem to a standard linear form that can be solved in a standard manner. The displacement field can then be calculated by using

$$\mathbf{u_h}(\xi, s) = \mathbf{N}(s) \sum_{i=1}^{n} c_i \xi^{-\lambda_i} \boldsymbol{\phi}_i \tag{2.25}$$

and the stress field by,

$$\sigma_{\mathbf{h}}(\xi, s) = \mathbf{D} \sum_{i=1}^{n} c_i \xi^{-\lambda_i - 1} \left[-\lambda_i \mathbf{B^1}(s) + \mathbf{B^2}(s) \right] \boldsymbol{\phi}_i \tag{2.26}$$

where $\mathbf{B^1}$ and $\mathbf{B^2}$ are given by

$$\mathbf{B^1}(s) = \mathbf{b^1}(s)\,\mathbf{N}(s) \tag{2.27}$$

$$\mathbf{B^2}(s) = \mathbf{b^2}(s)\,\mathbf{N}(s)_{,s} \tag{2.28}$$

and $\mathbf{b^1}$ and $\mathbf{b^2}$ are functions dependent only on the boundary definition (Deeks and Wolf, 2002).

2.5 Meshless methods

The boundary element method presented briefly in section 2.3 demonstrated that reducing the dimension of the problem by one, from volume integrals (as in the FEM) to surface integrals, can make the meshing process computationally simpler. This is important for both an initial analysis and in subsequent reanalysis where the geometry of the problem has been perturbed in some manner and as such the mesh will need to be updated.

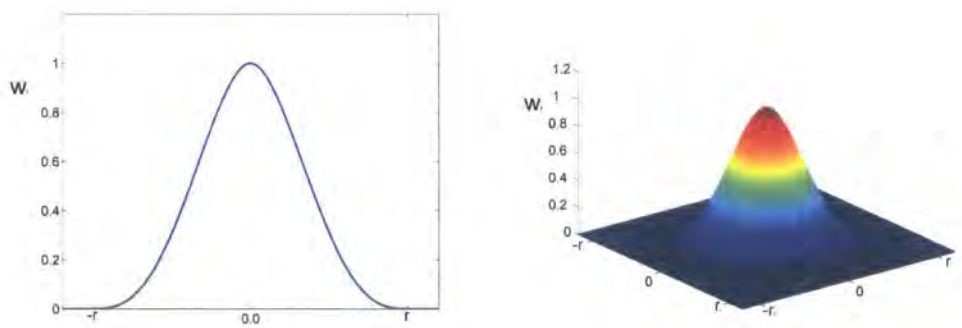
Meshless methods remove the need for a *mesh* around the problem and instead use nodes placed across the domain. As there are nodes in the problem it is still necessary to link the nodes to nodes situated near-by such that information about

the variables of concern can be propagated. However, as there is no mesh in the same sense as the finite or boundary element method it is necessary to use other techniques to ensure connectivity. The technique used in most meshless methods is that of a moving least squares approximation to displacements. It is assumed that the variable of interest will vary as a polynomial around each node in the domain. The influence from each of the nodes will overlap with nodes in the near vicinity and as a result the value of the shape function at a particular node is obtained by the weighted summation of these influences.

Typically simple polynomial basis functions are employed and these are weighted by using a symmetric weight function based on a spline or Gaussian function. Figure 2.10 shows an example of a 1-dimensional (figure 2.10(a)) and 2-dimensional (figure 2.10(b)) weighting function.

$$u^h = \sum_{i=1}^{N} \phi_i \hat{u}_i \tag{2.29}$$

where u^h is the approximated displacement, ϕ are shape functions and $\hat{u}_i$ is the fictitious displacement at node i.



(a) 1-dimensional function

(b) 2-dimensional function

Figure 2.10: Weighting functions for meshless methods

As equation (2.29) produces a moving least squares approximation to the displacement, u^h will not be the same as $\hat{u}_i$ but will approximate it. As a result it can be difficult to apply displacement boundary conditions to meshless methods as they are defined in terms of $\hat{u}_i$ and not u^h.

Calculation of the shape functions used within meshless methods is achieved by

noting that,

$$u^h = \mathbf{p}^T \mathbf{a} \tag{2.30}$$

where $\mathbf{p}$ is a vector containing the basis function terms and $\mathbf{a}$ is a vector of coefficients that are unknown. To calculate the coefficients in $\mathbf{a}$ we minimise the function,

$$J = \sum_{i=1}^{N} w_i \left[\mathbf{p}^T \mathbf{a} - \hat{u}_i\right]^2 \tag{2.31}$$

The shape functions can then be derived as

$$\Phi = \mathbf{p}^T \mathbf{A}^{-1} \mathbf{B} \tag{2.32}$$

where

$$\mathbf{A} = \sum_{i=1}^{N} w_i \mathbf{p}_i \mathbf{p}_i^T \tag{2.33}$$

$$\mathbf{B} = [w_1 \mathbf{p}_1, w_2 \mathbf{p}_2, \ldots, w_n \mathbf{p}_n] \tag{2.34}$$

It is necessary to derive a weak form of equilibrium and compatibility equations and this can be achieved via the use of a test function. The element free Galerkin (EFG) (Belytschko *et al.*, 1994) meshless method proposes a global weak form such that the integrations are performed over the complete problem domain, similar to the finite element method. This causes a problem as the method is no longer a *true* meshless method as it is necessary to incorporate integration cells within the problem domain and as a result of this a cell generator is required.

An alternative technique to the EFG method is the meshless local Petrov-Galerkin method (MLPG) (Atluri and Zhu, 1998) in which the the aim is not to satisfy the global weak form directly as in the EFG method. The MLPG aims to satisfy the weak form on a local level around each node. However, if these *zones* overlap throughout the domain and cover the entire domain as a set of zones the weak form will be satisfied for the global problem.

As a result of the meshless methods presented employing a moving least squares approximation for the displacement field it is not possible to directly apply displacement boundary conditions as used in other numerical methods. This is a result that u^h is an approximation to $\hat{u}_i$. Thus it is necessary to use penalty methods or Lagrange multipliers to impose the conditions (Augarde and Deeks, 2005).

2.6 Concluding remarks

Each of the methods presented in this section are extremely powerful in their own right, be it the adaptability of the finite element method or the ease of re-meshing a problem under the boundary element method and as such each method has its place in the engineer's toolbox.

For the purpose of this work the boundary element method will be considered as the ease of remeshing a problem is extremely important in the process of re-analysis. The boundary element method is preferred over meshless methods as the prescription of boundary conditions in the latter is problematic and thus increases the complexity of the method. However, parts of the work; in particular the section on preconditioning (section 7) could be applied to alternate numerical methods that generate matrix equations of the same form.

CHAPTER 3

Boundary Element Method

Following consideration of a number of numerical methods (chapter 2) the use of the boundary element method (BEM) has been found to have certain advantageous characteristics for a wide variety of problem types. The boundary element method has been applied to problems as diverse as fracture mechanics (Mukherjee, 1982), where the stress field becomes singular at the crack tip, to problems involving infinite domains for which the BEM is particularly suited since it is only only necessary discretise the boundary of the domain (Wu, 2000). Finally the BEM is suited to problems of reanalysis. As the boundary is the only part of the analysis model required to be discretised it is a relatively quick and easy operation to remesh the problem after a perturbation (Trevelyan and Wang, 2001a). Other benefits of the BEM for this particular type of problem will be discussed within the derivation.

This chapter will present the theory behind the boundary element method for elasticity, heat transfer and acoustics.

3.1 Derivation of the boundary element method

The boundary element method will be formulated for a general problem, as shown in figure 3.1, where the problem domain is indicated by Ω, with boundary Γ. Conditions are typically only prescribed on the boundary Γ but it is possible to deal with body forces using a variety of techniques.

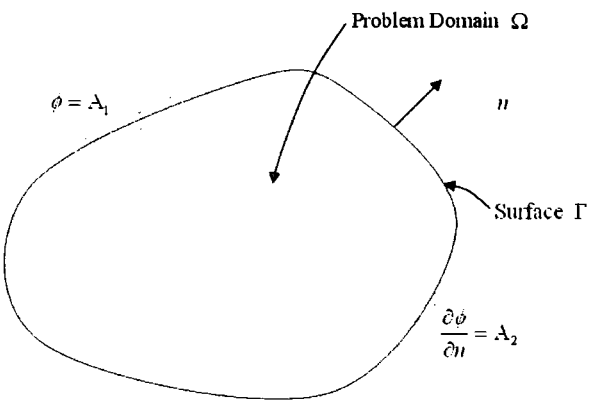


Figure 3.1: General boundary value problem

Various forms of condition can be applied on the boundary,

- A prescribed value of unknown function; for example $\phi = A_1$
- A prescribed value of the normal derivative of the unknown function ϕ; for example $\frac{\partial \phi}{\partial n} = A_2$
- A prescribed relationship between ϕ and $\frac{\partial \phi}{\partial n}$, e.g. a linear spring

Boundary conditions of the first type are called *Dirichlet conditions*, conditions of the second type are called *Neumann conditions* and the third type of condition is a *Robin condition*. There are further conditions that are non-linear in nature that can be prescribed on the domain such as contact conditions, as a result of the non-linear nature these require special treatment as discussed by Aliabadi (2002).

We can write integral equations that relate the boundary functions, ϕ and $\frac{\partial \phi}{\partial n}$, over the solution domain. It is possible to apply certain integral transformations to these problems; for example Green's Theorem (equation (3.1)),to reduce the

dimensionality by one such that volume integrals become surface integrals.

$$\int_{\Omega} \nabla \cdot \mathbf{f} \mathrm{d}\Omega = \int_{\Gamma} \mathbf{f} \cdot \mathbf{n} \mathrm{d}\Gamma \tag{3.1}$$

where $\mathbf{f}$ is an arbitrary vector function and $\mathbf{n}$ is the unit outward pointing normal. Another useful form of Green's Theorem is Green's second identity (equation (3.2)) which is used in many BE applications to reduce the dimensionality by one.

$$\int_{\Omega} \left(\phi \nabla^2 \lambda - \lambda \nabla^2 \phi\right) \mathrm{d}\Omega = \int_{\Gamma} \left(\phi \frac{\partial \lambda}{\partial n} - \lambda \frac{\partial \phi}{\partial n}\right) \mathrm{d}\Gamma \tag{3.2}$$

We will now look at the derivation of the boundary integral equations for specific types of problems.

3.1.1 Elasticity

To form the boundary integral equation for elasticity directly *Betti's reciprocal work theorem* will be applied. Considering a small element of the domain Ω we can form the equations of equilibrium

$$\sigma_{ij,j} + b_i = 0 \tag{3.3}$$

where σ_{ij} is a stress component and b_i the body force vector per-unit volume. From this we can state the weighted residual form

$$\int_{\Omega} \left(\sigma_{ij,j} + b_i\right) u_i^* \mathrm{d}\Omega = \int_{\Omega} \left(\sigma_{ij,j} u_i^* + b_i u_i^*\right) \mathrm{d}\Omega = 0 \tag{3.4}$$

where u_i^* is an arbitrary weighting function that will be defined completely later in the derivation. To allow the application of Green's theorem (equation (3.1)) the first term in the weighted residual form needs to be expanded by the use of the product rule (equation (3.5)).

$$\left(f\left(x\right) g\left(x\right)\right)' = f\left(x\right) g'\left(x\right) + f'\left(x\right) g\left(x\right) \tag{3.5}$$

Application of the product rule in differentiation (equation (3.5)) produces

$$\int_{\Omega} \left[\left(\sigma_{ij} u_i^*\right)_{,j} - \sigma_{ij} u_{i,j}^* + b_i u_i^*\right] \mathrm{d}\Omega = 0 \tag{3.6}$$

It should be noted that as $\left(\sigma_{ij} u_i^*\right)_{,j}$ is the divergence of $\left(\sigma_{ij} u_i^*\right)$ then we can now apply Green's theorem (equation (3.1)) to reduce the volume integral to a surface

integral

$$\int_{\Gamma} \sigma_{ij} u_i^* n_j \mathrm{d}\Gamma - \int_{\Omega} \sigma_{ij} u_{i,j}^* \mathrm{d}\Omega + \int_{\Omega} b_i u_i^* \mathrm{d}\Omega = 0 \tag{3.7}$$

where n_j are the components of the unit normal on the boundary Γ. Application of the *Cauchy stress transformation*

$$t_i = \sigma_{ij} n_j \tag{3.8}$$

produces

$$\int_{\Gamma} t_i u_i^* \mathrm{d}\Gamma - \int_{\Omega} \sigma_{ij} u_{i,j}^* \mathrm{d}\Omega + \int_{\Omega} b_i u_i^* \mathrm{d}\Omega = 0 \tag{3.9}$$

This statement is now equivalent to the principle of virtual work (PVW).

An equivalent statement to equation (3.9) can be produced where a fictitious load case does work on the real displacements

$$\int_{\Gamma} t_i^* u_i \mathrm{d}\Gamma - \int_{\Omega} \sigma_{ij}^* u_{i,j} \mathrm{d}\Omega + \int_{\Omega} b_i^* u_i \mathrm{d}\Omega = 0 \tag{3.10}$$

Application of Hooke's law* shows that $\sigma_{ij} u_{i,j}^* = \sigma_{ij}^* u_{i,j}$. Hence we can state

$$\int_{\Gamma} t_i u_i^* \mathrm{d}\Gamma + \int_{\Omega} b_i u_i^* \mathrm{d}\Omega = \int_{\Gamma} t_i^* u_i \mathrm{d}\Gamma + \int_{\Omega} b_i^* u_i \mathrm{d}\Omega \tag{3.11}$$

This is *Betti's reciprocal theorem.* Betti's reciprocal theorem is an extremely important theorem as it allows the relation between two physical systems from their respective displacement fields. This is advantageous if one of the displacement fields can be obtained relatively conveniently.

Currently, we have not defined the fictitious load case. By defining the fictitious load case to be the Dirac delta function,

$$\Delta(x - \kappa) = \begin{cases} \infty & \text{if } x = \kappa \\ 0 & x \neq \kappa \end{cases} \tag{3.12}$$

we can use the following property to remove one of the volume integrals,

$$\int_{\Omega} \Delta(x - \kappa) \mathrm{d}\Omega = 1 \tag{3.13}$$

*Remembering that $u_{i,j} = \varepsilon_{ij}$ and that the constitutive fourth order tensor is symmetric we can state $\sigma_{ij} u_{i,j}^* = C_{ijkl} \varepsilon_{kl} \varepsilon_{ij}^* = C_{ijkl} \varepsilon_{ij}^* \varepsilon_{kl} = C_{klij} \varepsilon_{ij}^* \varepsilon_{kl} = \sigma_{kl}^* u_{k,l}$

Thus the fictitious load case is represented by equation (3.14).

$$\sigma_{ij,j} + \Delta(x - \kappa) e_i(x) = 0 \tag{3.14}$$

where κ is the sample point under consideration and e_i is a unit vector in the direction of i. These properties allow the right hand side of equation (3.11) to be reduced to

$$\int_\Gamma t_i^* u_i d\Gamma + \int_\Omega b_i^* u_i d\Omega = \int_\Gamma t_i^* u_i d\Gamma + \int_\Omega \Delta(x - \kappa) e_i u_i d\Omega \tag{3.15}$$

The first term on the right hand side of equation (3.15) is as required but the second term can now be reduced due to the properties of the Dirac delta function that we have chosen as the arbitrary load case. Recalling that the integral of the Dirac delta function is unity (equation (3.13)) we can state that

$$\int_\Omega \Delta(x - \kappa) e_i u_i d\Omega = e_i u_i(\kappa) \tag{3.16}$$

The term e_i is a unit vector in the direction of i and the term u_i is a displacement in the same direction thus,

$$\int_\Gamma t_i^* u_i d\Gamma + \int_\Omega \Delta(x - \kappa) e_i u_i d\Omega = \int_\Gamma t_i^* u_i d\Gamma + e_i u_i(\kappa) \tag{3.17}$$

Hence equation (3.11) can now be written as

$$u_i(\kappa) e_i + \int_\Gamma t_i^* u_i d\Gamma = \int_\Gamma t_i u_i^* d\Gamma + \int_\Omega b_i u_i^* d\Omega \tag{3.18}$$

By setting the boby force vector to be the Dirac delta function u^* is given by the displacement solution to equation (3.14) known as the Kelvin solution (Thomson, 1848). The displacement solution is related to the free space Green's function by

$$u_i^* = U_{ij} e_j \tag{3.19}$$

For 2-dimensional plane strain problems the fundamental solution is given by,

$$U_{ij} = C_1 \left\{ C_2 \ln\left[\frac{1}{r}\right] \delta_{ij} + r_{,i} r_{,j} \right\} \tag{3.20}$$

where $r = |x - \kappa|$, x is the field point, δ_{ij} is the Kronecker delta given by equation (3.21) and C_1 and C_2 are functions of material properties defined in equation (3.22).

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{3.21}$$

$$C_1 = \frac{1}{8\pi\mu(1-\nu)}, \qquad C_2 = (3-4\nu) \tag{3.22}$$

where μ is the material shear modulus and ν is the Poisson's ratio.

In the case of plane stress we can obtain the fundamental solution by using modified material properties.

$$\begin{aligned} \nu' &= \tfrac{\nu}{1+\nu} \\ E' &= \tfrac{E(1+2\nu)}{(1+\nu)^2} \\ \mu' &= \mu \end{aligned} \tag{3.23}$$

For 3-dimensional problems the displacement solution is given by,

$$U_{ij} = \frac{C_1}{2}\frac{1}{r}\left[C_2\delta_{ij} + r_{,i}r_{,j}\right] \tag{3.24}$$

For axi-symmetric problems the displacement solutions are more complicated, the interested reader can find these in either Becker (1992) or Aliabadi (2002).

We can obtain the traction fundamental solution, t_i^* by differentiating the displacement fundamental solution and applying Hooke's law. Similarly to the displacement case

$$t_i^* = T_{ij}e_j \tag{3.25}$$

Thus, for 2-dimensional plane strain,

$$T_{ij} = C_3\left(\frac{1}{r}\right)\left\{r_{,n}\left[C_4\delta_{ij} + 2r_{,i}r_{,j}\right] - C_4\left[r_{,j}n_i - r_{,i}n_j\right]\right\} \tag{3.26}$$

where $\mathbf{n}$ is the unit normal and C_3 and C_4 are functions of material properties given by equation (3.27)

$$C_3 = \frac{-1}{4\pi(1-\nu)}, \qquad C_4 = (1-2\nu) \tag{3.27}$$

For 3-dimensional problems the traction fundamental solution is given by,

$$T_{ij} = \frac{C_3}{2}\frac{1}{r^2}\left[r_{,n}\left(C_4\delta_{ij} + 3r_{,i}r_{,j}\right) + c_4\left(r_{,j}n_i - r_{,i}n_j\right)\right] \tag{3.28}$$

For axi-symmetric problems the traction solutions are more complicated, the interested reader can find these in Becker (1992).

Thus, equation (3.18) can be written as

$$u_i(\kappa) + \int_\Gamma T_{ij}u_j\mathrm{d}\Gamma = \int_\Gamma U_{ij}t_j\mathrm{d}\Gamma + \int_\Omega U_{ij}b_j\mathrm{d}\Omega \tag{3.29}$$

Equation (3.29) contains a volume integral on the right hand side due to body forces in the real load case. Body forces complicate the derivation of the boundary integral equation due to the techniques required deal with the volume integral. There are a number of techniques that can be employed to compute the volume integral (Aliabadi, 2002). The simplest method is the introduction of integration cells throughout the domain though this raises the problem that the method is no longer a *proper* boundary only method.

The Galerkin vector approach involves the introduction of a new function called the Galerkin tensor, G_{ij}. The Galerkin tensor can be related to the fictitious load case for linear stress analysis by equation (3.30).

$$U_{ij} = G_{ij,kk} - C_{GV} G_{ij,kj} \tag{3.30}$$

where C_{GV} is a constant based on material properties given by,

$$C_{GV} = \frac{1}{2\left(1-\nu\right)}$$

Equation (3.30) can be substituted into the body force integral in equation (3.29).

$$\int_{\Omega} U_{ij} b_j \mathrm{d}\Omega = \int_{\Omega} b_j \left(G_{ij,kk} - C_{GV} G_{ij,kj}\right) \mathrm{d}\Omega \tag{3.31}$$

Applying Green's second identity reduces the volume integral to a surface only integral. For example, a body under acceleration will have a constant body force hence b_i can be taken outside of the integral.

$$\int_{\Omega} b_j \left(G_{ij,kk} - C_{GV} G_{ij,kj}\right) \mathrm{d}\Omega = b_j \int_{\Omega} \left(G_{ij,kk} - C_{GV} G_{ij,kj}\right) \mathrm{d}\Omega \tag{3.32}$$

$$= b_j \int_{\Gamma} \left(G_{ij,k} - C_{GV} G_{ij,j}\right) n_k \mathrm{d}\Gamma \tag{3.33}$$

The Galerkin vector is given by equation (3.34) for 2-dimensional problems and equation (3.35) for 3-dimensional problems.

$$G_{ij} = \frac{r^2}{8\pi G} \ln\left(\frac{1}{r}\right) \delta_{ij} \tag{3.34}$$

$$G_{ij} = \frac{r}{8\pi G} \delta_{ij} \tag{3.35}$$

where G is the shear modulus.

A more complicated approach is to employ dual reciprocity techniques which are applicable to general forms of body force.

The Dual Reciprocity Method (DRM) (Nardini and Brebbia, 1983) involves the approximation of the body forces within the volume integral,

$$b_j \simeq \sum_{k=1}^{N+L} \alpha_i^k f\left(\mathbf{x}^k, \mathbf{x}\right) \tag{3.36}$$

where α_i^k are a set of initially unknown coefficients, $f\left(\mathbf{x}^k, \mathbf{x}\right)$ are approximating functions, N is the total number of boundary nodes in the discretisation, and L is the total number of internal nodes. The expansion given in equation (3.36) is a global expansion as it is valid over the whole domain. Substituting equation (3.36) into the body force volume integral in equation (3.29) produces

$$\int_\Omega U_{ij} b_j \mathrm{d}\Omega = \sum_{k=1}^{N+L} \alpha_i^k \int_\Omega U_{ij} f^k \mathrm{d}\Omega \tag{3.37}$$

The choice of f^k can be problem dependant, however, a common choice of function is

$$f^k = c + r\left(\mathbf{x^k}, \mathbf{x}\right) \tag{3.38}$$

where c is a constant and $r\left(\mathbf{x^k}, \mathbf{x}\right)$ is the distance between the points $\mathbf{x^k}$ and $\mathbf{x}$. The second integral in equation (3.37) can be substituted by,

$$\int_\Omega U_{ij} f^k \mathrm{d}\Omega = u_j^k(\kappa) + \int_\Gamma T_{ij} u_j^k \mathrm{d}\Gamma - \int_\Gamma U_{ij} t_j^k \mathrm{d}\Gamma \tag{3.39}$$

where u_j^k and t_j^k are particular solutions which are known functions when f^k is defined. Substituting this into equation (3.29) produces

$$u_i(\kappa) + \int_\Gamma T_{ij} u_j \mathrm{d}\Gamma - \int_\Gamma U_{ij} t_j \mathrm{d}\Gamma = \sum_{k=1}^{N+L} \alpha_i^k \left[u_j^k(\kappa) + \int_\Gamma T_{ij} u_j^k \mathrm{d}\Gamma - \int_\Gamma U_{ij} t_j^k \mathrm{d}\Gamma \right] \tag{3.40}$$

The solution of the right hand side can be found in a similar manner to the left hand side. In this work we will assume that there are no body forces in the real case i.e. $b_j = 0$, for all j and as such equation (3.29) can now be arranged as,

$$u_i(\kappa) + \int_\Gamma T_{ij} u_j \mathrm{d}\Gamma = \int_\Gamma U_{ij} t_j \mathrm{d}\Gamma \tag{3.41}$$

Equation (3.41) now has one term that is not dependent only on the boundary, u_i. To create a completely boundary only solution we simply enforce the condition that u_i must lie on the boundary such that $\kappa \in \Gamma$.

Equation (3.41) can now be integrated around the boundary Γ. Care is required in the integration phase as the integrations involving the fundamental solutions, equations (3.20) and (3.26) have terms of the form $\frac{1}{r^\alpha}$ where $\alpha = 1, 2$ as a result when integrating these functions the integrals will be either weakly or strongly singular. As the U_{ij} integral is weakly singular of the logarithmic form shown in (3.20) this can be integrated using a logarithmic Gauss quadrature scheme (outlined in section 4.3.2). The T_{ij} term, however, needs to be integrated in the Cauchy principal value sense.

To perform the integration we surround the collocation point κ by a semi-circle of radius r_ϵ, the integration is now performed in three parts considering the limit as $r_\epsilon \to 0$. The case considered is represented in figure 3.2.

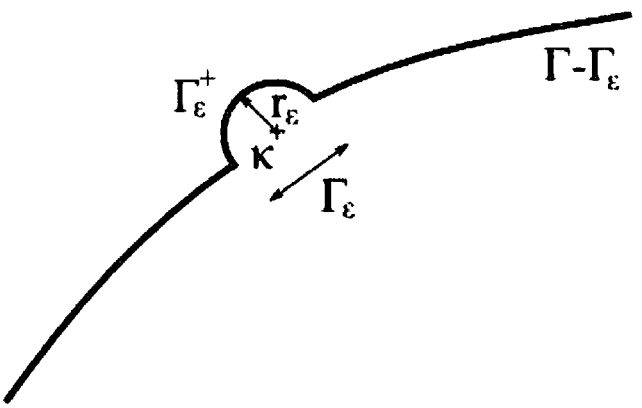


Figure 3.2: Integration in the Cauchy principal value sense

$$\begin{aligned}\int_\Gamma T_{ij}u_j d\Gamma = &\lim_{r_\epsilon \to 0}\left\{\int_{\Gamma-\Gamma_\epsilon} T_{ij}u_j d\Gamma\right\} \\ &+ \lim_{r_\epsilon \to 0}\left\{\int_{\Gamma_\epsilon^+} T_{ij}\left(u_j\left(x\right) - u_j\left(\kappa\right)\right) d\Gamma\right\} \\ &+ u_j\left(\kappa\right)\lim_{r_\epsilon \to 0}\left\{\int_{\Gamma_\epsilon^+} T_{ij} d\Gamma\right\}\end{aligned} \tag{3.42}$$

The first term in equation (3.42) is the Cauchy principal value, the second term will

vanish since as $r_\epsilon \to 0$ then $u_i(x) \to u_i(\kappa)$, and the final term can be simplified to

$$u_j(\kappa) \lim_{r_\epsilon \to 0} \left\{ \int_{\Gamma_\epsilon^+} T_{ij} \mathrm{d}\Gamma \right\} = \alpha_{ij}(\kappa) u_j(\kappa) \tag{3.43}$$

where α can be considered to be representative of the boundary. For a smooth boundary $\alpha_{ij}(\kappa) = -\frac{\delta_{ij}}{2}$. Thus, equation (3.41) can be rewritten as,

$$c_{ij}(\kappa) u_i(\kappa) + \int_\Gamma T_{ij} u_j \mathrm{d}\Gamma = \int_\Gamma U_{ij} t_j \mathrm{d}\Gamma \tag{3.44}$$

where $c_{ij}(\kappa)$ varies between 0 and 1. Typical values of c_{ij} are represented in figure 3.3. Equation (3.44) is referred to as the boundary integral equation (BIE) and is the fundamental building block behind the boundary element method.

(a) $c_{ij}(\kappa) = 0.25$ (b) $c_{ij}(\kappa) = 0.50$ (c) $c_{ij}(\kappa) = 0.75$

Figure 3.3: Values for $c_{ij}(\kappa)$

To allow the integration of equation (3.44) it is necessary to discretise the model into individual elements (figure 3.4). As a result the BIE can now be written in its discrete form,

$$c_{ij}(\kappa) u_i(\kappa) + \sum_{elem} \int_{\Gamma_e} T_{ij} u_j \mathrm{d}\Gamma_e = \sum_{elem} \int_{\Gamma_e} U_{ij} t_j \mathrm{d}\Gamma_e \tag{3.45}$$

where Γ_e is the boundary for the element being integrated. Shape functions are applied to the elements to allow the extraction of displacements and tractions at discrete locations. Shape functions are typically defined for elements with either 2 or 3 nodes per element for 2-dimensional analysis. For 2 nodes per element,

$$N_1(\xi) = \frac{1-\xi}{2} \tag{3.46}$$

$$N_2(\xi) = \frac{1+\xi}{2} \tag{3.47}$$

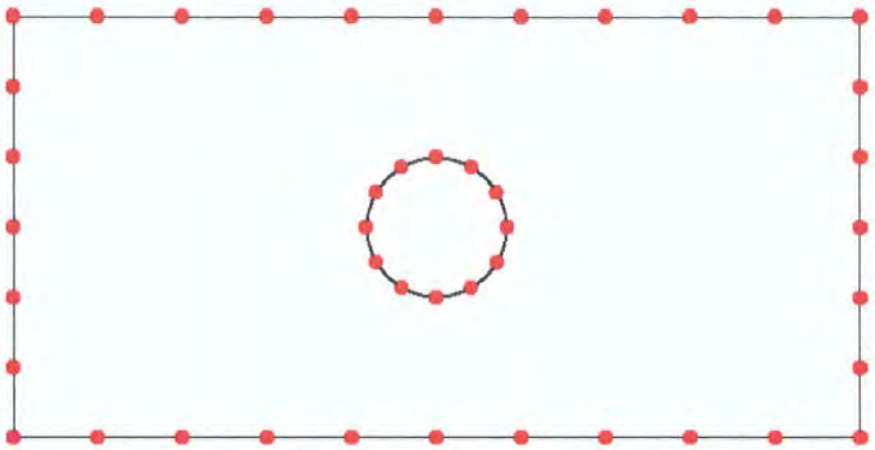


Figure 3.4: Discretised sample problem

and for 3 nodes per element,

$$N_1(\xi) = \frac{-\xi}{2}(1-\xi) \tag{3.48}$$

$$N_2(\xi) = 1-\xi^2 \tag{3.49}$$

$$N_3(\xi) = \frac{\xi}{2}(1+\xi) \tag{3.50}$$

where ξ is a local coordinate such that the element is bounded by $[-1,+1]$ (figure 3.5). In this work elements will have 3 nodes. Figure 3.6 shows the variation of shape

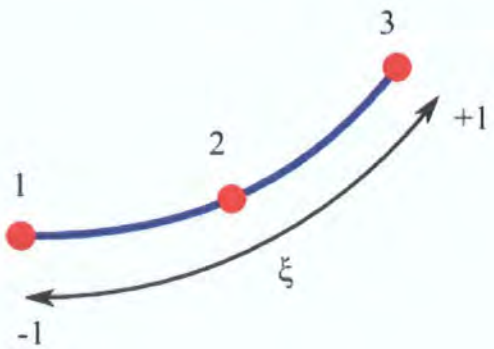


Figure 3.5: Sample element

functions for elements with three nodes. Similar shape functions can be derived for 3-dimensional problems but involve a second local coordinate ζ.

Shape functions are used for interpolation of the nodal quantities along the element length. Equation (3.51) shows the use of shape functions for the interpolation

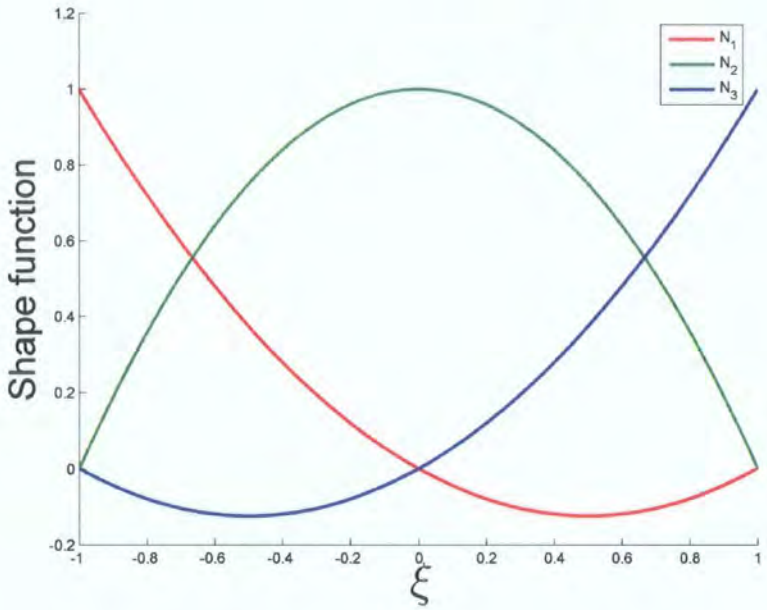


Figure 3.6: Shape functions for elements with 3 nodes

of nodal displacements along an element.

$$u(\xi) = N_1(\xi)\, u_1 + N_2(\xi)\, u_2 + N_3(\xi)\, u_3 = \mathbf{Nu} \tag{3.51}$$

An iso-parametric element uses the same shape functions for interpolating the coordinates of the nodal locations along the element.

The introduction of a local coordinate system allows the use of Gauss quadrature schemes of integration (see chapter 4 for more details). As a result it is necessary to alter the integration to reflect the change in variable and incorporate a Jacobian. The Jacobian is defined as,

$$J(\xi) = \frac{\partial \mathbf{x}}{\partial \xi} \tag{3.52}$$

Thus, the discretised form of the BIE becomes,

$$c_{ij}(\kappa)\, u_i(\kappa) + \sum_{elem} \int_{\Gamma_e} T_{ij} N_i J(\xi)\, \mathrm{d}\xi u_j = \sum_{elem} \int_{\Gamma_e} U_{ij} N_i J(\xi)\, \mathrm{d}\xi t_j \tag{3.53}$$

where u_j and t_j contain nodal values of the displacement and traction respectively.

Equation (3.53) can now be integrated numerically (see chapter 4 for more details) by placing the source point at each node and integrating across each element

in turn. Integration at the first node results in the following equation,

$$c_1 u_1 + h_{11} u_1 + h_{12} u_2 + \cdots + h_{1n} u_n = g_{11} t_1 + g_{12} t_2 + \cdots + g_{1n} t_n \tag{3.54}$$

where h and g contain the value of the integrals containing t^* and u^* respectively. Repeating the integration around the boundary leads to a linear set of equations,

$$\begin{aligned}
c_1 u_1 + h_{11} u_1 + h_{12} u_2 + h_{13} u_3 + h_{14} u_4 + \ldots &= g_{11} t_1 + g_{12} t_2 + g_{13} t_3 + g_{14} t_4 + \ldots \\
c_2 u_2 + h_{21} u_1 + h_{22} u_2 + h_{23} u_3 + h_{24} u_4 + \ldots &= g_{21} t_1 + g_{22} t_2 + g_{23} t_3 + g_{24} t_4 + \ldots \\
c_3 u_3 + h_{31} u_1 + h_{32} u_2 + h_{33} u_3 + h_{34} u_4 + \ldots &= g_{31} t_1 + g_{32} t_2 + g_{33} t_3 + g_{34} t_4 + \ldots \\
c_4 u_4 + h_{41} u_1 + h_{42} u_2 + h_{43} u_3 + h_{44} u_4 + \ldots &= g_{41} t_1 + g_{42} t_2 + g_{43} t_3 + g_{44} t_4 + \ldots
\end{aligned}$$

These can be combined into the standard matrix equation.

$$\mathbf{Hu} = \mathbf{Gt} \tag{3.55}$$

where $\mathbf{H}$ and $\mathbf{G}$ are square matrices containing the respective influence coefficients, $\mathbf{u}$ and $\mathbf{t}$ are vectors for the displacements and tractions respectively around the boundary. Figure 3.7 shows the form of the matrices after the integration stage with unknown terms edged with black. Currently, as no boundary conditions have been applied both the $\mathbf{u}$ and $\mathbf{t}$ vectors are entirely unknown. Application of boundary

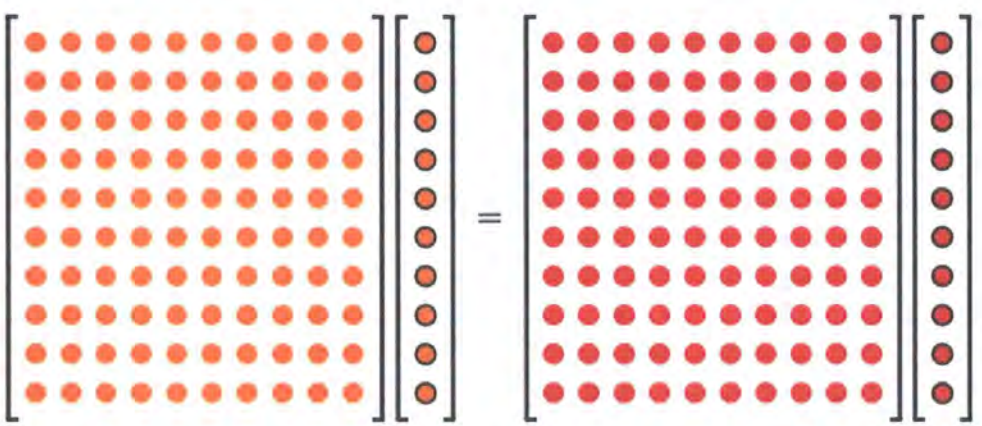


Figure 3.7: Matrix terms - $\mathbf{Hu} = \mathbf{Gt}$

conditions (figure 3.8) fills n terms within the unknown vectors such that the matrix system consists of n equations with n unknown terms. To allow the problem in

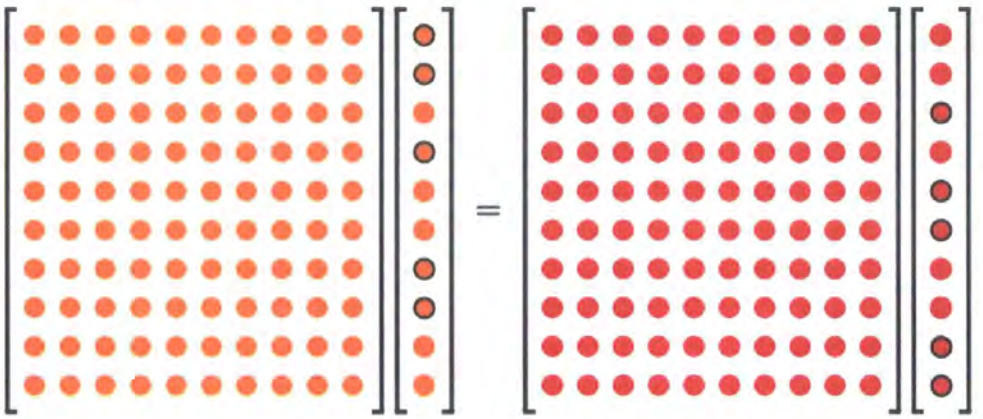


Figure 3.8: Matrix terms - Boundary conditions applied

figure 3.8 to be solved it is necessary to move all of the unknown terms in the vectors to the left hand side vector such that all of the terms on the right hand side are known (figure 3.9). Multiplying the right hand side matrix-vector product

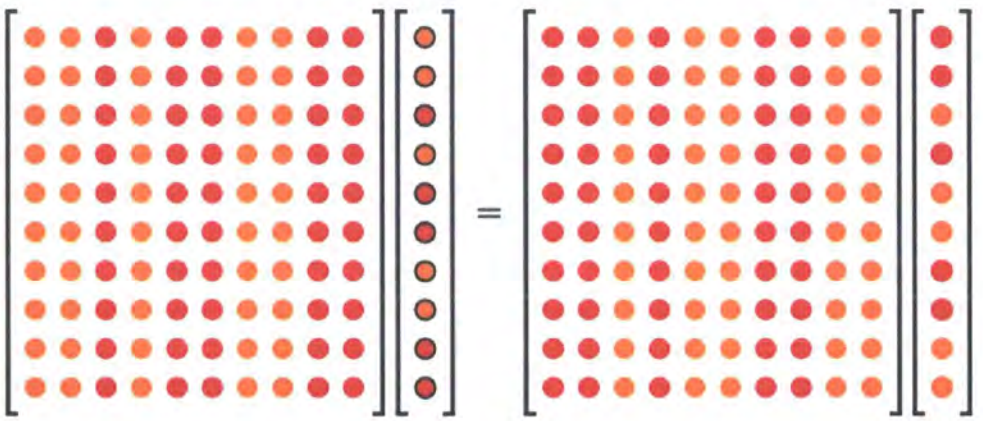


Figure 3.9: Matrix terms - Matrix rows and columns exchanged

produces a vector that is completely known (figure 3.10), producing the well known linear system ready for solution.

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{3.56}$$

where the $\mathbf{A}$ is an $n \times n$ matrix of known values, $\mathbf{b}$ is a known vector of size n and $\mathbf{x}$ are the unknown tractions and displacements to be solved for. Equation (3.56) can now be solved using a direct or iterative solver (see chapter 5 for more details).

Once the boundary solution has been calculated we can apply the boundary

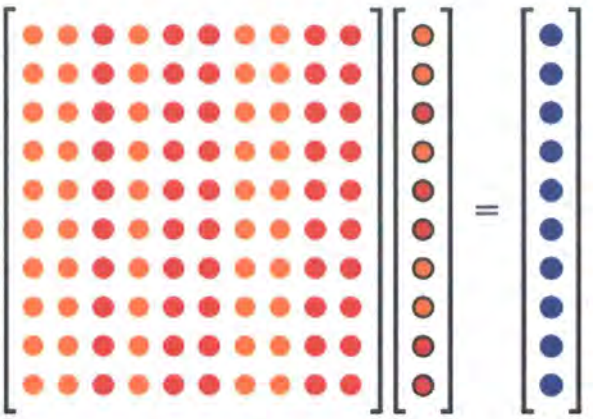


Figure 3.10: Matrix terms - $\mathbf{Ax} = \mathbf{b}$

integral equation to calculate displacements at points internal to the domain Ω. Moving the point κ within the domain and applying equation (3.44) we can calculate the corresponding displacement throughout the domain allowing the production of accurate contour plots of displacement. It should be noted that with points internal to the domain $c_{ij}(\kappa) = 1$ due to the integrations not involving singularities, as such the integrals can be performed using conventional Gauss-Legendre quadrature.

To calculate stresses we can differentiate the boundary integral equation, equation (3.41), and then apply Hooke's law to produce the stress (σ) form of the boundary integral equation.

$$\sigma_{ij} + \int_{\Gamma} S_{kij} u_k \mathrm{d}\Gamma = \int_{\Gamma} D_{kij} t_k \mathrm{d}\Gamma \tag{3.57}$$

where S_{kij} and D_{kij} are third order tensors.

$$S_{kij} = C_5 \left(\frac{1}{r}\right)^2 \begin{bmatrix} n_i \left[2\nu r_{,j} r_{,k} + C_4 \delta_{jk}\right] + n_j \left[2\nu r_{,i} r_{,k} + C_4 \delta_{ik}\right] \\ +n_k \left[2C_4 r_{,i} r_{,j} - (1 - 4\nu)\, \delta_{ij}\right] \\ +2r_{,n} \left[C_4 \delta_{ij} r_{,k} + \nu \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j}\right) - 4 r_{,i} r_{,j} r_{,k}\right] \end{bmatrix} \tag{3.58}$$

$$D_{kij} = -C_3 \left(\frac{1}{r}\right) \left[C_4 \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j} - \delta_{ij} r_{,k}\right) + 2 r_{,i} r_{,j} r_{,k}\right] \tag{3.59}$$

where C_5 is a constant based on material properties given by,

$$C_5 = \frac{\mu}{2\pi (1 - \nu)} \tag{3.60}$$

Equation (3.57) can typically be integrated using a standard Gauss-Legendre quadrature scheme. Care does need to be taken if the internal points are located in the vicinity of the boundary as the third order tensors D_{kij} and S_{kij} will be strongly singular and hyper-singular integrals respectively when integrated.

3.1.2 Potential flow

For the problem of potential flow we need to consider *Laplace's equation*,

$$\nabla^2\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = 0 \tag{3.61}$$

where ∇^2 is the Laplacian operator and ϕ is the potential function, temperature in heat transfer problems, whilst x and y are Cartesian coordinates. Laplace's equation is similar to Poisson's equation except that the right hand side in the Poisson equation is non-zero, typically a function of ϕ.

The fundamental solution of Laplace's equation is based on the three-dimensional solution to a concentrated potential

$$\lambda(\kappa, Q) = \frac{1}{2\pi}\ln\left[\frac{1}{r(\kappa, Q)}\right] \tag{3.62}$$

where r is as defined for elasticity.

To reduce this problem by one dimension, to make it a boundary-only problem, we must apply Green's second identity, equation (3.2), with ϕ representing the unknown potential and λ the fundamental solution to Laplace's equation, equation (3.61).

$$\int_{\Omega} \left(\phi\nabla^2\lambda - \lambda\nabla^2\phi\right) \mathrm{d}\Omega = \int_{\Gamma} \left(\phi\frac{\partial\lambda}{\partial n} - \lambda\frac{\partial\phi}{\partial n}\right) \mathrm{d}\Gamma \tag{3.63}$$

The potential function ϕ satisfies $\nabla^2\phi = 0$ everywhere, by definition. The fundamental solution, λ however, satisfies $\nabla^2\lambda = 0$ everywhere except at the point κ itself, as the function is singular. To allow for this singularity we surround the point κ by a circle of radius r_ϵ and then take the limit as $r_\epsilon \to 0$. Thus, equation (3.63) can now be re-written as,

$$\int_{\Omega-\Omega_\epsilon} \left(\phi\nabla^2\lambda - \lambda\nabla^2\phi\right) \mathrm{d}\Omega = \int_{\Gamma+\Gamma_\epsilon} \left(\phi\frac{\partial\lambda}{\partial n} - \lambda\frac{\partial\phi}{\partial n}\right) \mathrm{d}\Gamma \tag{3.64}$$

As we have now eliminated the point κ from this equation we can see that,

$$\nabla^2\phi = 0 \qquad \text{and} \qquad \nabla^2\lambda = 0$$

thus the left hand side of equation (3.64) becomes zero, and the right hand side surface integral can be split into two separate surface integrals to be dealt with separately.

$$0 = \int_{\Gamma} \left(\phi\frac{\partial\lambda}{\partial n} - \lambda\frac{\partial\phi}{\partial n}\right) \mathrm{d}\Gamma + \int_{\Gamma_\epsilon} \left(\phi\frac{\partial\lambda}{\partial n} - \lambda\frac{\partial\phi}{\partial n}\right) \mathrm{d}\Gamma \tag{3.65}$$

To calculate the integral on the boundary Γ_ϵ we can use the angle α and substitute $\mathrm{d}\Gamma = r_\epsilon \mathrm{d}\alpha$, additionally we can note that,

$$\frac{\partial\lambda}{\partial n} = \frac{\partial\lambda}{\partial r}\frac{\partial r}{\partial n} = \frac{1}{2\pi}\frac{-1}{r}(-1) = \frac{1}{2\pi r}$$

Consideration of the second integral of equation (3.65) as we take the limit $r_\epsilon \to 0$ within the limits $0 < \alpha < 2\pi$,

$$\int_{\Gamma_\epsilon} \left(\phi\frac{\partial\lambda}{\partial n} - \lambda\frac{\partial\phi}{\partial n}\right) \mathrm{d}\Gamma = \frac{1}{2\pi}\int_0^{2\pi} \left[\phi\left(\frac{1}{r_\epsilon}\right) - \ln\left(\frac{1}{r_\epsilon}\right)\frac{\partial\phi}{\partial n}\right] r_\epsilon \mathrm{d}\alpha \tag{3.66}$$

$$= \frac{1}{2\pi}\int_0^{2\pi} \left[\phi + (r_\epsilon \ln r_\epsilon)\frac{\partial\phi}{\partial n}\right] \mathrm{d}\alpha \tag{3.67}$$

$$= \frac{1}{2\pi}(2\pi\phi) = \phi \tag{3.68}$$

Substitution of equation (3.66) into equation (3.65) produces the potential form of the BIE,

$$\phi(\kappa) + \int_{\Gamma} K_1(\kappa, Q)\,\phi(Q)\,\mathrm{d}\Gamma(Q) = \int_{\Gamma} K_2(\kappa, Q)\frac{\partial\phi(Q)}{\partial n}\mathrm{d}\Gamma(Q) \tag{3.69}$$

where K_1 and K_2 are the kernels such that,

$$K_1(\kappa, Q) = \frac{\partial\lambda(\kappa, Q)}{\partial n} \tag{3.70}$$

$$K_2(\kappa, Q) = \lambda(\kappa, Q) = \frac{1}{2\pi}\ln\left[\frac{1}{r(\kappa, Q)}\right] \tag{3.71}$$

Similarly to the elasticity problem, section 3.1.1, we move the point κ to the boundary to make the solution a full *boundary-only* solution and introduce the jump term $c(\kappa)$ which is dependent on boundary geometry, see figure 3.3,

$$c(\kappa)\,\phi(\kappa) + \int_{\Gamma} K_1(\kappa, Q)\,\phi(Q)\,\mathrm{d}\Gamma(Q) = \int_{\Gamma} K_2(\kappa, Q)\frac{\partial\phi(Q)}{\partial n}\mathrm{d}\Gamma(Q) \tag{3.72}$$

It should be noted that the kernels for potential problems, equations (3.70) and (3.71), are entirely functions of geometry.

3.1.3 Acoustics

The boundary element method is ideally suited for problems of acoustics as it is only necessary to mesh the boundary. As a result if the problem domain can be considered to be infinite, for example the water waves around the legs of an oil rig in the ocean (figure 3.11) can be modelled by the Helmholtz equation in water of constant depth. Then the boundary element method merely meshes the legs whereas the finite element method would require a *false* exterior boundary to be included limiting how far the results are calculated for.

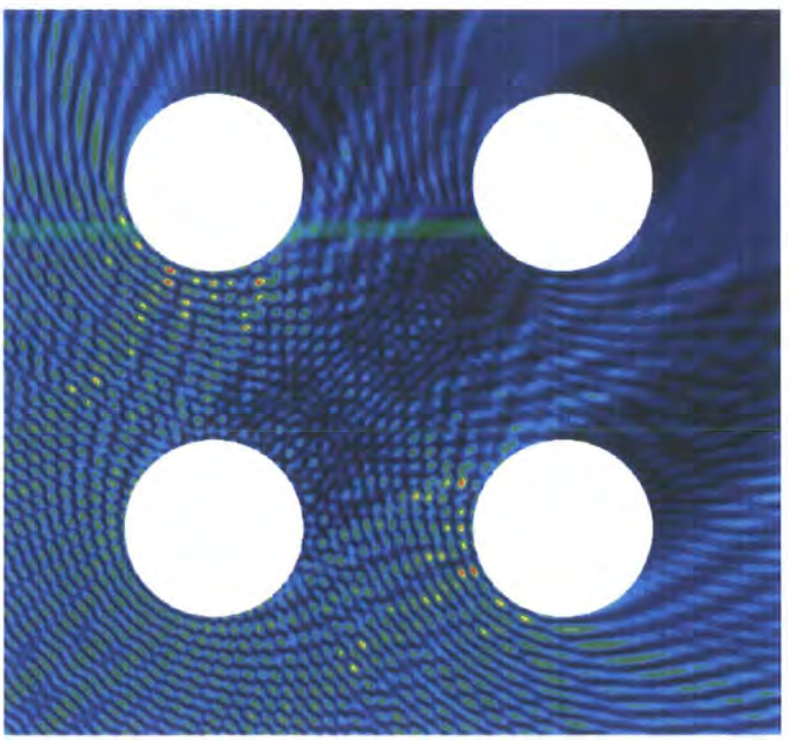


Figure 3.11: Example of an infinite domain (Perrey-Debain *et al.*, 2004; Trevelyan, 2006)

In acoustics we can assume that the time variation of response will be of the form $e^{-i\omega t}$. Then the wave equation will reduce to the well known Helmholtz equation

$$\nabla^2\psi + k^2\psi = 0 \tag{3.73}$$

This can be expressed in boundary integral form as

$$c(\kappa)\,\phi(\kappa) + \int_{\Gamma} \frac{\partial G(\kappa, Q)}{\partial n}\phi(Q)\,\mathrm{d}\Gamma(Q) = \int_{\Gamma} G(\kappa, Q)\,\frac{\partial \phi(Q)}{\partial n}\mathrm{d}\Gamma(Q) + \phi_i \tag{3.74}$$

where $\mathrm{Re}(c(\kappa))$ is related to the boundary shape and varies between 0 and 1 and $\mathrm{Im}(c(\kappa)) = 0$, ϕ_i is the incident wave. For this problem it can be shown that the fundamental solution is given by

$$G = \frac{i}{4}H_0(kr) \tag{3.75}$$

$$\frac{\partial G}{\partial n} = \frac{k}{4}\frac{\partial r}{\partial n}\left[Y_1(kr) - iJ_1(kr)\right] \tag{3.76}$$

where H_n is the Hankel function which can be calculated as,

$$H_n(kr) = J_n(kr) - iY_n(kr) \tag{3.77}$$

where J_n is the Bessel function of the first kind and Y_n is the Bessel function of the second kind.

$$J_n(x) = x^n \sum_{m=0}^{\infty} \frac{(-1)^m x^{2m}}{2^{2m+n} m!\,(n+m)!} \tag{3.78}$$

$$Y_n(x) = \frac{2}{\pi}J_n(x)\left(\ln\frac{x}{2} + \gamma\right) + \frac{x^n}{\pi}\sum_{m=0}^{\infty}\frac{(-1)^{m-1}(h_m + h_{m+n})}{2^{2m+n} m!\,(m+n)!}x^{2m} - \frac{x^{-n}}{\pi}\sum_{m=0}^{n-1}\frac{(n-m-1)!}{2^{2m-n} m!}x^{2m} \tag{3.79}$$

where γ is the *Euler constant* given by

$$\gamma = \lim_{n\to\infty}\left(\sum_{k=1}^{n}\frac{1}{k} - \ln n\right) = 0.57721566\ldots \tag{3.80}$$

and h_m is given by

$$h_0 = 0 \tag{3.81}$$

$$h_m = \sum_{i=0}^{m}\frac{1}{i} \tag{3.82}$$

For 3-dimensional problems the fundamental solution of the Helmholtz equation is given by,

$$G = \frac{e^{ikr}}{4\pi r} \tag{3.83}$$

Thus, equation (3.74) can now be integrated by substitution of the fundamental solutions. Care is required when integrating the Bessel functions as they contain singularities that need to be integrated using logarithmic Gauss quadrature (see chapter 4 for more details).

For acoustic problems the mesh density is constrained such that,

$$\text{Element Length} < \frac{\lambda}{4}$$

This ensures that the problem can be fully captured by the elements.

Additionally, for problems involving infinite domains it is possible to suffer from non-uniqueness at the eigenfrequencies of the associated internal problem. Thus it is necessary to employ additional techniques to overcome this non-uniqueness such as CHIEF (Schenck, 1968). CHIEF involves the collocation at a number of points inside the body as additional constraint equations.

3.2 Concluding remarks

In this chapter the boundary element method has been presented for three different numerical problems. In each case fundamental solutions have been presented.

Details on the integration procedure have been discussed with relation to the building of matrix equations, the application of appropriate boundary conditions and the subsequent reduction to the well known linear system

$$\mathbf{Ax} = \mathbf{b} \tag{3.84}$$

which can be solved using either a direct or iterative solver (see chapter 5).

The boundary element method has a number of advantages and disadvantages. Disadvantages of the boundary element method include the use of complicated and singular integral equations and the use of fundamental solutions to create a boundary formulation. The method is not suitable for thin shell analysis. This is because of the large surface/volume ratio and hence the distance between collocation points and elements becomes small. This can cause inaccuracies within the numerical integrations.

The main advantages are that it is a boundary only numerical method and as a result it is particularly good for problems of reanalysis. This is a consequence of the ease with which a mesh can be generated as it is not necessary to triangulate throughout the volume. The matrix equations generated are smaller than comparable numerical methods. However, the matrices are fully populated and thus sparse techniques can not be applied.

The BEM is also known for its accuracy and efficiency in problems such as fracture mechanics, where the solution contains singularities. Additionally, the BEM is extremely efficient for large problems containing areas of geometric detail because of the requirement to only mesh the boundary. Moreover, this benefit can be extended to problems involving infinite domains such as acoustic or wave problems.

CHAPTER 4

Numerical Integration Techniques

The boundary integral equation, repeated in equation (4.1) for completeness from chapter 3, contains complicated integrals

$$c_{ij}(\kappa)\, u_i(\kappa) + \int_\Gamma T_{ij} u_j \mathrm{d}\Gamma = \int_\Gamma U_{ij} t_j \mathrm{d}\Gamma \tag{4.1}$$

Due to the complicated nature of the equation it can only be integrated analytically for simple cases and as a result a numerical integration scheme is typically used to approximate the integrals. In this chapter a number of different numerical integration techniques will be presented and considered.

4.1 Newton-Cotes

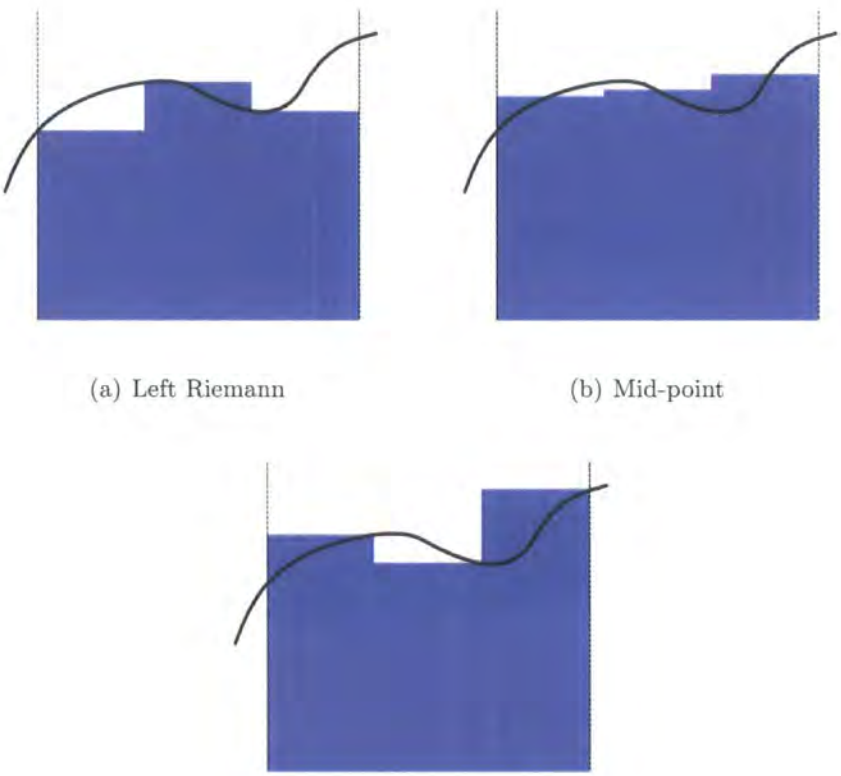
Newton-Cotes integration is the generic name for an integration technique that involves taking the value of the function to be integrated at equally spaced points and weighting it. More commonly it is used to refer to techniques that involve tabular data at fixed points. More details on these techniques can be found within any mathematical textbook, for example Kreyszig (1999) or Chapra and Canale (2002).

4.1.1 Rectangular integration

The simplest form of Newton-Cotes integration is rectangular integration. This technique involves the splitting of the integration into equal width strips and fitting a constant; order zero, function across the strip. There are three ways of deciding the height of the strips,

1. Left Riemann approximation
2. Midpoint approximation
3. Right Riemann approximation

Figure 4.1 shows each of the splitting techniques for rectangular integration.



(a) Left Riemann

(b) Mid-point

(c) Right Riemann

Figure 4.1: Splitting techniques for rectangular integration

Rectangular integration can be represented by equation (4.2). The width of the strips used determines the accuracy of the integration.

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=1}^{n} f\left(a + k'\frac{b-a}{n}\right)\frac{b-a}{n} + \mathcal{E} \tag{4.2}$$

where $\mathcal{E}$ is the error term and

$$k' = \begin{cases} k-1 & \text{if left Riemann approx.} \\ k-\frac{1}{2} & \text{if midpoint approx.} \\ k & \text{if right Riemann approx.} \end{cases}$$

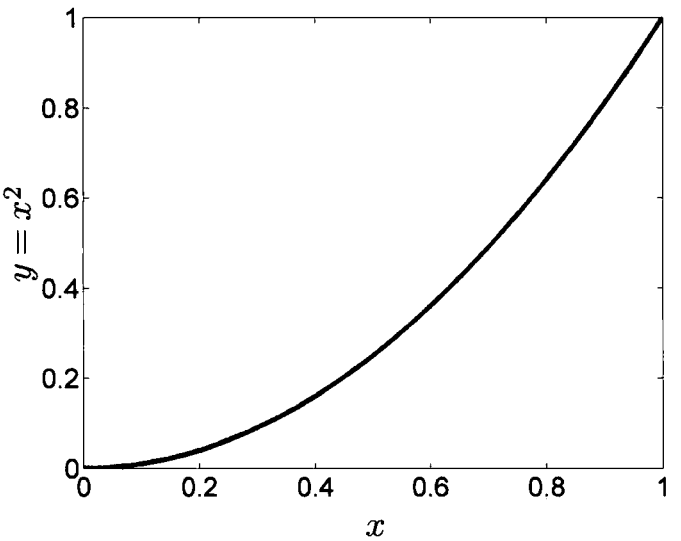


Figure 4.2: Example integration

Table 4.1 shows how the accuracy of integrating $f(x) = x^2$; displayed in figure 4.2, varies with strip count between the limits of 0 and +1. The accuracy will be determined against the analytical solution given in equation (4.3).

$$\int_0^{+1} x^2\mathrm{d}x = \left[\frac{1}{3}x^3\right]_0^{+1} = \frac{1}{3} \tag{4.3}$$

Table 4.1 shows that the midpoint splitting technique is the most accurate method of rectangular integration for this particular integral. This is a result of the gradient of the equation being integrated. Figure 4.3 shows application of the midpoint technique and the approximate cancellation areas above and below the strip.

Strips	Left Riemann		Midpoint		Right Riemann	
	Int. Value	Error (%)	Int. Value	Error (%)	Int. Value	Error (%)
1	0.0000	-100.00	0.2500	-25.00	1.0000	200.00
2	0.1250	-62.50	0.3125	-6.25	0.6250	87.50
3	0.1852	-44.44	0.3241	-2.77	0.5185	55.55
4	0.2188	-34.36	0.3281	-1.57	0.4688	40.64
5	0.2400	-28.00	0.3300	-1.00	0.4400	32.00
6	0.2546	-23.62	0.3310	-0.70	0.4213	26.39
7	0.2653	-20.41	0.3316	-0.52	0.4082	22.46
8	0.2734	-17.98	0.3320	-0.40	0.3984	19.52
9	0.2798	-16.06	0.3323	-0.31	0.3909	17.27
10	0.2850	-14.50	0.3325	-0.25	0.3850	15.50

Table 4.1: Rectangular integration of $\int_0^{+1} x^2 \mathrm{d}x$

The midpoint integration is an example of an *open* Newton-Cotes scheme, i.e. it does not sample at the ends of the interval of integration. This is advantageous in some circumstances.

4.1.2 Trapezoidal integration

Trapezoidal integration is similar to rectangular integration as the integration area is split into strips in the same manner. A first order fit is then applied to the strip such that it becomes a trapezium as opposed to a rectangle, as indicated in figure 4.4.

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{b-a}{n}\left(\frac{f(a)+f(b)}{2}+\sum_{k=1}^{n-1} f\left(a+k\frac{b-a}{n}\right)\right)+\mathcal{E} \qquad (4.4)$$

An advantage of the trapezium rule is that it is possible to tell if the trapezium rule will be an over or under estimate of the true integral. If a function is always concave in nature ($\frac{d^2f}{dx^2} > 0$) then the trapezium rule will over estimate the integral, figure 4.5(a). If the function is convex in nature ($\frac{d^2f}{dx^2} < 0$) then the integral will be

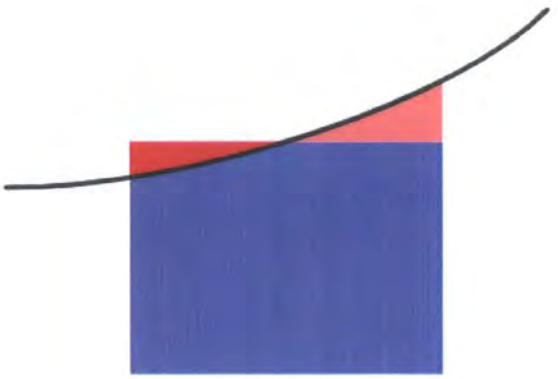


Figure 4.3: Application of midpoint rule for an individual strip

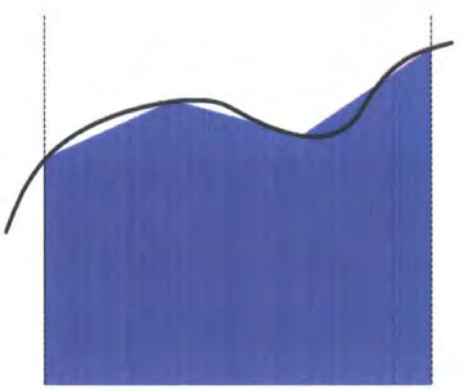
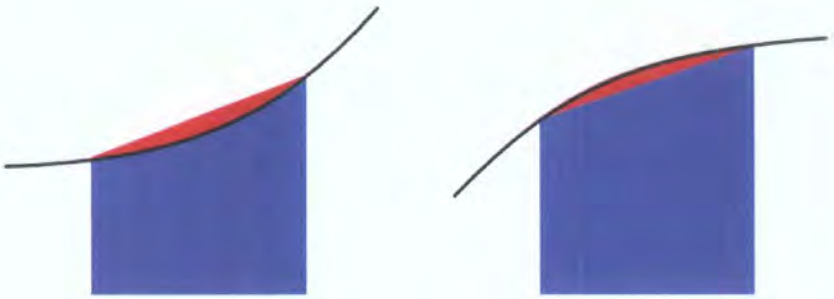


Figure 4.4: Application of trapezoid rule

an under estimate, figure 4.5(b). If the function contains an inflection point then the error will be harder to approximate a-priori.



(a) Concave function (b) Convex function

Figure 4.5: Error estimation for trapezoidal rule

As the trapezium rule is an order 1 fit to the function it is commonly seen to be more accurate than rectangular integration. Table 4.2 shows how the error varies when performing the integration given in equation (4.3).

Number of Strips	Integral Value	Error (%)
1	0.5000	50.00
2	0.3750	12.50
3	0.3519	5.57
4	0.3438	3.14
5	0.3400	2.00
6	0.3380	1.40
7	0.3367	1.01
8	0.3359	0.77
9	0.3354	0.62
10	0.3350	0.50

Table 4.2: Trapezoidal integration of $\int_0^{+1} x^2 \mathrm{d}x$

Comparing the results of table 4.2 with those in table 4.1 it can be seen that the trapezium rule is more accurate than both left and right Riemann integration. This

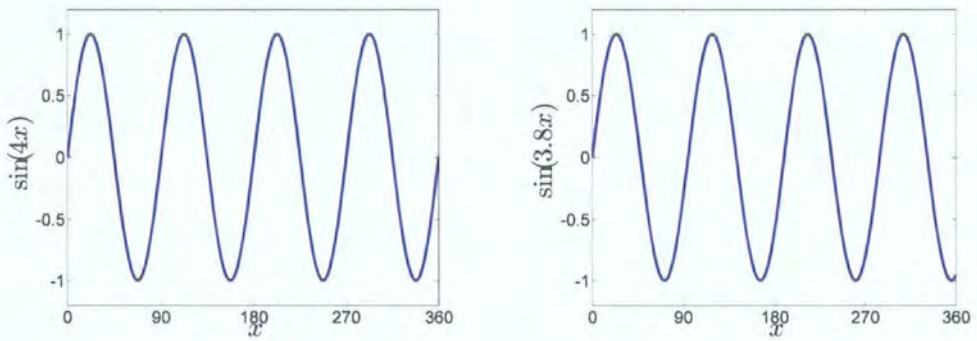
is on account of the higher order of fit that can be achieved when using the trapezium rule. Comparison with the midpoint rule, however, shows that rectangular integration is more accurate and this is a result of the cancellation effect demonstrated in figure 4.3.

The trapezoidal rule exhibits very rapid, exponential, convergence for cases in which the integrand is periodic in the interval over which the integral is performed. Figure 4.6 displays plots of equations (4.5) and (4.6) over the range $0° < x \leq 360°$.

$$f_1(x) = \sin(4\pi x) \tag{4.5}$$

$$f_2(x) = \sin(3.8\pi x) \tag{4.6}$$

Figure 4.7 compares the number of strips used in the integration with the absolute



(a) $f_1(x)$ (b) $f_2(x)$

Figure 4.6: Plots of equations (4.5) and (4.6)

error in the integral value.

4.1.3 Simpson's rule

Simpson's rule is a method of approximating $f(x)$ by a quadratic function, order 2.

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{h}{3}\left[f(a) + 2\sum_{k=1}^{\frac{n}{2}-1} f(a+2kh) + 4\sum_{k=1}^{\frac{n}{2}} f(a+(2k-1)h) + f(b)\right] + \mathcal{E} \tag{4.7}$$

where h is the strip width given by $h = \frac{b-a}{n}$.

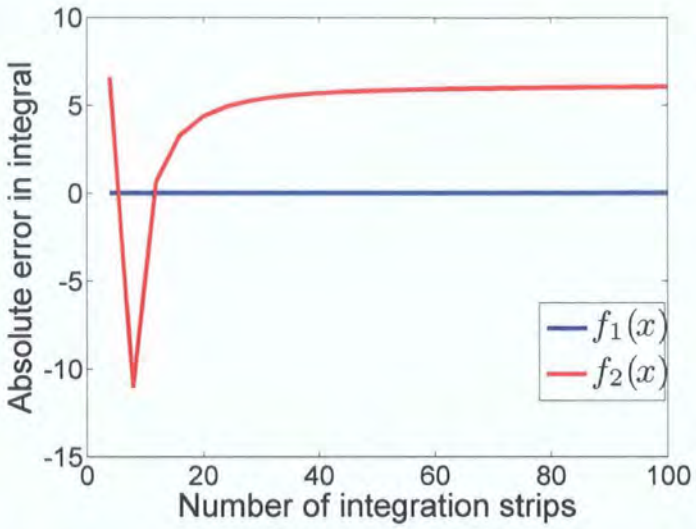


Figure 4.7: Absolute error in numerical integration as strip count is varied

Equation (4.7) shows the form of Simpson's rule, due to the quadratic fit the integration is more complicated. As Simpson's rule is a quadratic fit then it is a requirement of the method to have an even number of strips for the integration.

Table 4.3 compares Simpson's rule with both rectangular integration (using the midpoint rule) and the trapezium rule for the problem of,

$$\int_0^4 \left(x^4 - 6x^3 + 11x^2 - 8x + 7\right) \mathrm{d}x \tag{4.8}$$

which has the analytical solution

$$\begin{aligned} \int_0^4 \left(x^4 - 6x^3 + 11x^2 - 8x + 7\right) \mathrm{d}x &= \left[\frac{1}{5}x^5 - \frac{3}{2}x^4 + \frac{11}{3}x^3 - 4x^2 + 7x\right]_0^4 \\ &= 19.4667 \end{aligned} \tag{4.9}$$

Figure 4.8 shows how the function to be integrated varies over the region of interest.

Table 4.3 shows that Simpson's rule has a higher rate of convergence to the analytical solution then either rectangular integration or the trapezium rule. This is a result of the quadratic fit being able to fit the function much more accurately than either of the previous two techniques.

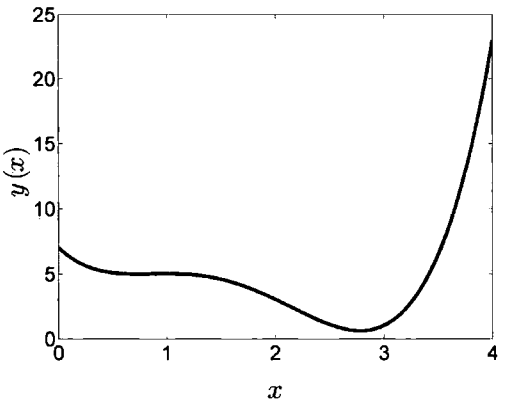


Figure 4.8: Plot of $f(x) = x^4 - 6x^3 + 11x^2 - 8x + 7$

Strips	Rectangle		Trapezium		Simpson's	
	Int. Value	Error (%)	Int. Value	Error (%)	Int. Value	Error (%)
2	12.0000	-38.36	36.0000	84.93	28.0000	43.84
4	17.2500	-11.39	24.0000	23.29	20.0000	2.74
6	18.4527	-5.21	21.5144	10.52	19.5720	0.54
8	18.8906	-2.96	20.6250	5.95	19.5000	0.17
10	19.0963	-1.90	20.2099	3.82	19.4803	0.07
12	19.2088	-1.32	19.9835	2.65	19.4733	0.03
14	19.2770	-0.97	19.8467	1.95	19.4702	0.02
16	19.3213	-0.75	19.7578	1.50	19.4688	0.01

Table 4.3: Comparison of Newton-Cotes techniques for integration

4.2 Transformation of integrals

Currently all of the integration techniques introduced have had arbitrary limits of a to b. This has been acceptable as the integration interval has been divided into separate strips and the appropriate scheme repeated over each of those strips.

It is necessary for certain integration schemes to work between fixed limits for the integral to be valid. Thus it is necessary to apply a transformation to the original integral such that it has more suitable limits.

$$\int_a^b f(x)\,\mathrm{d}x = \int_\alpha^\beta f[x(\xi)]\,J(\xi)\,\mathrm{d}\xi \tag{4.10}$$

where $J(\xi)$ is called the *Jacobian* and is given by,

$$J(\xi) = \frac{dx}{d\xi} \tag{4.11}$$

For double integrals the Jacobian is formed in a similar manner

$$J(\xi_1,\xi_2) = \frac{\partial(x,y)}{\partial(\xi_1,\xi_2)} = \begin{bmatrix} \frac{\partial x}{\partial \xi_1} & \frac{\partial x}{\partial \xi_2} \\ \frac{\partial y}{\partial \xi_1} & \frac{\partial y}{\partial \xi_2} \end{bmatrix} \tag{4.12}$$

and since $\mathrm{d}x = |J(\xi)|\,\mathrm{d}\xi$ the determinant of the Jacobian matrix in equation (4.12) is included in the integral in a similar fashion to equation (4.10).

4.3 Gauss quadrature

Gaussian quadrature is an alternative method of numerically integrating a function $f(x)$. Gaussian quadrature employs variable abscissae and weights and can be represented by equation (4.13).

$$\int_{-1}^{+1} f(x)\,\mathrm{d}x \approx \sum_{i=1}^{n} w_i f(x_i) \tag{4.13}$$

where w_i is the weight associated with the abscissa x_i this is performed over n Gauss points.

Equation (4.13) shows that the limits of integration are -1 to $+1$. As a result it will be necessary to employ a Jacobian of transformation to move from arbitrary limits of integration to the Gauss limits.

There are a number of techniques for calculating the position of the abscissae. Three main techniques will be presented in this thesis

1. Gauss-Legendre (including logarithmic Gauss-Legendre quadrature)
2. Gauss-Radau
3. Gauss-Lobatto

In most of the strategies presented abscissae are placed symmetrically about the origin.

4.3.1 Gauss-Legendre quadrature

Gauss-Legendre quadrature involves the distribution of n abscissae throughout the integration. To determine the location of the abscissae and their respective weights it is useful to start with low order cases before moving to higher orders of integration. Gauss-Legendre quadrature of order n is capable of integrating exactly a polynomial of order $2n-1$ (Pozrikidis, 1998).

Consideration of $n=2$ will allow the exact integration of a cubic function. To calculate the abscissae and weights it is necessary to consider the following cases, for which second order Gauss-Legendre quadrature will be exact,

$$f_3(x) = a_0x^3 + a_1x^2 + a_2x + a_3 \tag{4.14}$$

$$f_2(x) = b_0x^2 + b_1x + b_2 \tag{4.15}$$

$$f_1(x) = c_0x + c_1 \tag{4.16}$$

$$f_0(x) = d_0 \tag{4.17}$$

where

$$a_0 \neq 0, \qquad b_0 \neq 0, \qquad c_0 \neq 0, \qquad d_0 \neq 0$$

such that the order of the function being integrated is not altered. Considering equation (4.13), setting $n=2$ and assuming that $x_1 \neq x_2 \in [-1,+1]$. It is possible to show the following four relations,

$$2 = w_1 + w_2 \tag{4.18}$$

$$0 = x_1w_1 + x_2w_2 \tag{4.19}$$

$$\frac{2}{3} = x_1^2 w_1 + x_2^2 w_2 \tag{4.20}$$

$$0 = x_1^3 w_1 + x_2^3 w_2 \tag{4.21}$$

Equations (4.18), (4.19), (4.20) and (4.21) now define the weights and abscissae for the integration. Solving for the weights, w_i, and abscissae, x_i, yields

$$w_1 = w_2 = 1 \tag{4.22}$$

$$x_1 = -x_2 = \sqrt{\frac{1}{3}} \tag{4.23}$$

These are the well known $\pm\sqrt{\frac{1}{3}}$ seen in many finite and boundary element codes for Gauss-Legendre quadrature.

Gauss-Legendre quadrature can be extended for higher orders of integration allowing accurate fits for more complicated functions. To calculate the abscissae for the integrations it is possible to use the locations of the roots of the Legendre polynomial with order n (Pozrikidis, 1998). To calculate the associated weights, w_i, it is necessary to calculate (Davis and Rabinowitz, 1984),

$$w_i = -\frac{i_{n+1}}{i_n} \frac{1}{P_{n+1}(x_i) P_n'(x_i)} \tag{4.24}$$

where P_n is the Legendre polynomial of order n.

Relevant weights and abscissae tend to be stored within integration routines, however if a higher order of integration is required than stored it is possible to calculate weights and abscissae relatively cheaply (Press, 2002). Tables of abscissae and weights are available by numerous authors, for example (Pozrikidis, 1998) and (Abramowitz and Stegun, 2002).

4.3.2 Logarithmic Gauss quadrature

The boundary element method contains integrals with singularities of varying order. Equation (4.25) repeats one of the integrals from the displacement form of the boundary integral equation which has a weak singularity due to a logarithmic term.

$$\int_\Gamma U_{ij} t_j \mathrm{d}\Gamma = \int_\Gamma C_1 \left\{ C_2 \ln\left[\frac{1}{r}\right] \delta_{ij} + r_{,i} r_{,j} \right\} t_j \mathrm{d}\Gamma \tag{4.25}$$

To integrate this it is possible to use an alternative form of Gauss quadrature, which has a logarithmic weighting function,

$$\int_0^{+1} \ln\left(\frac{1}{x}\right) f(x)\,\mathrm{d}x \approx \sum_{i=1}^{n} w_i f(x_i) \tag{4.26}$$

It should be noted that the integral in equation (4.26) has limits of $[0, +1]$ and as such needs to be transformed appropriately. For cases in the BEM where the collocation point lies at the mid-node of an element, one can simply use equation (4.26) separately on the two halves of the element.

It is only necessary to use logarithmic Gauss quadrature to integrate the logarithmic term and the regular part of the integral ($C_1 r_{,i} r_{,j}$) can be integrated using equation (4.13).

Abramowitz and Stegun (2002) contains tables of weights and abscissae that can be used for the integration of logarithmic functions using this form of Gauss quadrature.

4.3.3 Gauss-Radau and Gauss-Lobatto quadrature

Gauss-Radau and Gauss-Lobatto integration are similar to Gauss-Legendre integration as they involve the placing of abscissae and associated weights throughout the integration domain.

Gauss-Radau integration fixes one of the abscissae at an endpoint ± 1 and then distributes the remaining $(n-1)$ abscissae symmetrically through the domain. This reduces the order of polynomial that can be integrated accurately to $2n-2$. Gauss-Lobatto integration, however involves setting abscissae at both end-points of the integration (-1 and $+1$) and then placing $(n-2)$ free abscissae through the integral area. This allows the accurate integration of polynomials of order $2n-3$.

The locations of the free abscissae are found in a similar manner to the abscissae for Gauss-Legendre quadrature. As one of the abscissa has been placed at one of the end nodes in Gauss-Radau quadrature it can be shown that the remaining abscissae are given by the roots of the $n-1$ Legendre polynomial, as a further abscissa is fixed in Gauss-Lobatto it is necessary to find the roots of the $n-2$ Legendre polynomial (Pozrikidis, 1998).

4.3.4 Order of integration

A number of authors have considered the order of Gauss integration required for particular types of integration problem. As Gauss-Legendre quadrature is designed to be accurate for polynomials of order $2n-1$, integrating functions such as those contained in the boundary element method requires generic rules to be created based on parameters within the integral.

Typically authors consider the generic cases of integrals of $\mathcal{O}\left(\frac{1}{x^m}\right)$ where $m > 1$. These can then be used as guidelines for the integrals from numerical methods such as the boundary element method. Eberwien *et al.* (2005) considered integrals around a single element and calculated the required Gauss-Legendre order to ensure that the error was kept below 0.1%. Furthermore they compared this result with previously generated tables for Gauss orders by Jun *et al.* (1985) and Bu and Davies (1995).

By employing a variable order scheme as proposed by Eberwien *et al.* (2005) it is possible to optimise the computational effort required to compute an integral whilst ensuring that a certain degree of accuracy is maintained within the integral.

4.4 Singular integrals

The evaluation of singular integrals is a necessary problem when using the boundary element method as it is a requirement to integrate the element that the collocation point lies within. As a result a number of authors have considered this problem.

For weakly singular integrals it is possible to employ specific integration schemes such as logarithmic Gauss quadrature (section 4.3.2) but for higher orders of singularity this cannot be used. Telles (1987) devised a technique involving a polynomial transformation to improve accuracy in numerical evaluation of singular and near-singular integrals.

Two techniques were proposed by Telles (1987). The first technique employed a second order polynomial transformation of the form,

$$\eta(\gamma) = a\gamma^2 + b\gamma + c \tag{4.27}$$

where the following requirements are met,

$$\left.\frac{\partial \eta}{\partial \gamma}\right|_{\bar{\eta}} = 0 \tag{4.28}$$

$$\eta(1) = 1 \tag{4.29}$$

$$\eta(-1) = -1 \tag{4.30}$$

where $\bar{\eta}$ is the value of η at a singularity whose effect we seek to minimise. For near-singular integrals it will be location on the element closest to the singularity. From these restrictions it can be seen that the coefficients a, b and c are given by the following,

$$a = -c \tag{4.31}$$

$$b = 1 \tag{4.32}$$

$$c = \frac{\bar{\eta} \pm \sqrt{(\bar{\eta}^2 - 1)}}{2} \tag{4.33}$$

This transformation can be applied only when $-1 \leq \bar{\eta} \leq 1$. However this restriction limits the applicability of the transformation. A more complicated third order polynomial transformation can be employed to remove the restriction on location of the singularity.

$$\eta(\gamma) = a\gamma^3 + b\gamma^2 + c\gamma + d \tag{4.34}$$

In addition to the conditions applied to the second order polynomial, it is necessary to define an additional parameter,

$$\left.\frac{\partial^2 \eta}{\partial \gamma^2}\right|_{\bar{\eta}} = 0 \tag{4.35}$$

A solution to this problem is given by the following coefficients

$$a = \frac{1}{Q} \tag{4.36}$$

$$b = -\frac{3\bar{\gamma}}{Q} \tag{4.37}$$

$$c = \frac{3\bar{\gamma}^2}{Q} \tag{4.38}$$

$$d = -b \tag{4.39}$$

$$Q = 1 + 3\bar{\gamma}^2 \tag{4.40}$$

where $\bar{\gamma}$ is the value of γ such that $\eta(\bar{\gamma}) = \bar{\eta}$.

By applying the polynomial transformation it is necessary to introduce a Jacobian, the Jacobian introduced will have the property that it passes through zero at the point $\bar{\eta}$ as a result this cancels the singularity that occurs at this point.

The application of these polynomial transformations increases the computational complexity of the integration process but allows the accurate integration of the small proportion of integrals that contain singularities. An additional benefit of the third order polynomial transformation is that it can be applied to an integral without a singularity without any loss of accuracy. Telles (1987) investigated this effect when considering near-singular integrals.

Higher order singularities rely on an alternative approach such as the method proposed by Kutt (1975). Consider the Hadamard finite part integral,

$$I = \oint_s^r \frac{g(x)}{(x-s)^k} \mathrm{d}x \tag{4.41}$$

It is possible to represent I by a scalar product of the form

$$(r-s)^{1-k} \sum_{i=1}^{N} g(x_i) \left[w_i^{(k)} + \frac{c_i^{(k)} \ln|r-s|}{(k-1)!} \right] \tag{4.42}$$

where $w_i^{(k)}$ are the weights at N equispaced points, $x_i \in [s, r]$ and $c_i^{(k)}$ are the coefficients for the $(k-1)$ numerical derivative of g at the origin. For the case where the singularity lies within the integral limits, equation (4.43), it is possible to split the integral into two finite-part integrals, equation (4.44).

$$I = \oint_a^b \frac{x-s}{|x-s|^{k+1}} f(x) \mathrm{d}x \tag{4.43}$$

$$\oint_a^b \frac{x-s}{|x-s|^{k+1}} f(x) \mathrm{d}x = (-1)^{k+1} \oint_a^s \frac{f(x)}{(x-s)^k} \mathrm{d}x + \oint_s^b \frac{f(x)}{(x-s)^k} \mathrm{d}x \tag{4.44}$$

However, as the strategy proposed by Kutt is developed for the case of $\oint_s^r$ where $(r > s)$ it is necessary to substitute $x = -y$ for the first finite-part integral.

$$\oint_a^b \frac{x-s}{|x-s|^{k+1}} f(x) \mathrm{d}x = -\oint_{-s}^{-a} \frac{f(-x)}{[x-(-s)]^k} \mathrm{d}x + \oint_s^b \frac{f(x)}{(x-s)^k} \mathrm{d}x \tag{4.45}$$

It is now possible to apply equation (4.42) to both of the finite-part integrals.

4.5 Exact integration

Zhang and Zhang (2004a) have considered the use of exact integration techniques and applied them to a variety of singular and hyper-singular integrals which are commonly found within the boundary element method. Consideration of this problem for individual elements allows a single integration routine to be implemented regardless of the singularity due to the singularity being implicitly included in the integration.

Unfortunately the freedom that this technique provides is computationally expensive and the routine required for integrating the relevant functions is not suitable for the real-time analysis at which this work is aimed.

4.6 Concluding remarks

In this section a number of integration techniques have been presented. The main techniques currently used in boundary element codes are those of a Gauss quadrature nature, in particular Gauss-Legendre quadrature and logarithmic Gauss quadrature.

To deal with the singular and hyper-singular integrals it is necessary to combine high order Gauss-Legendre integration schemes along with a technique to remove the singularity, such as the schemes proposed by Telles (1987) and Kutt (1975).

CHAPTER 5

Equation Solution Techniques

Numerical integration of the boundary integral equation for elasticity, equation (3.41), heat transfer, equation (3.72), produces the following matrix equation

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{5.1}$$

where the vector $\mathbf{x}$ contains the unknown displacements and tractions for the boundary problem. We can use two main types of technique to solve for the unknowns, a direct solver or an iterative solver. A number of direct and iterative solvers will be discussed in this chapter highlighting the advantages and disadvantages of each.

5.1 Direct solvers

Using a direct solver ensures that a solution will be reached as long as the matrix $\mathbf{A}$ is not singular. The drawback of this method is that it is a fixed procedure and as a result can be time-consuming, particularly for large sets of equations. As the procedure is fixed the solution time does not depend on the condition number or form of the matrix.

The two main direct solvers employed in the solution of equations such as equa-

tion (5.1) are

- Gaussian Elimination
- Lower-Upper (LU) factorisations

These two methods are related.

5.1.1 Gaussian elimination

Gaussian elimination is a simple method of reducing the square matrix to its upper triangular form. This can then be used to solve for any **b** vector, as such it is exceptionally quick at evaluating multiple solutions to problems which only vary in the **b** vector as it only requires a back-substitution for each solution. This situation occurs within the BEM for multiple load cases in which the type of boundary condition is the same but different in magnitude. The Gaussian elimination algorithm is shown (in pseudo-code) in algorithm (5.1).

Algorithm 5.1: Gaussian elimination of $\mathbf{Ax} = \mathbf{b}$

```
1: for j = 1 to (N − 1) do // Loop over the matrix rows
2:   for i = 0 to (j − 1) do // Loop over the matrix columns
3:     if a_ii = 0 then
4:       return "Error: Singular Matrix"
5:     else
6:       F = a_ij / a_ii // Calculate the scale factor
7:       for k = 0 to (N − 1) do
8:         a_jk = a_jk − (F a_ik)
9:       end for
10:      b_j = b_j − (F b_i)
11:    end if
12:  end for
13: end for
```

After application of algorithm (5.1) the matrix **A** is an upper triangular matrix

which can be solved via a simple back-substitution of the **b** vector to extract the unknowns in **x**.

The algorithm can be written in a number of ways, simply by differing the order of the i, j and k loops. One problem with algorithm (5.1) is that although it will always produce a solution (assuming that the matrix is not singular) the solution can be inaccurate due to numerical rounding. This problem can be reduced by the use of partial pivoting (Kreyszig, 1999).

The scheme for partial pivoting is listed in algorithm (5.2) Partial pivoting is

Algorithm 5.2: Partial pivoting

1: **if** $|a_{ii}| < threshold$ **then** // Where threshold is typically 10^{-6} for our cases
2: Find row l below i where $a_{li} = \max |a_{mi}|$ and $a_{li} \neq 0$
3: Swap rows l and i in matrix **A** and vector **b**
4: **end if**

effective at ensuring that accuracy is maintained within the solution phase, whilst increasing the computational cost of the method by a minimal amount. An alternative is to employ full pivoting in which the threshold is not checked and rows are swapped regardless, this is acceptable for algorithms produced in C++ due to the efficient ability to swap the rows within the matrix.

The full partial-pivoted Gaussian elimination algorithm is listed in algorithm (5.3). This algorithm has a computational cost of $\frac{2}{3}N^3$ where N is the number of equations to be solved.

5.1.2 LU factorisation

The process of calculating an LU factorisation is similar to Gaussian elimination discussed in section (5.1.1). It differs in that the end product is two triangular matrices; one lower and one upper triangular. These can then be solved by a forward and then back-substitution process (algorithm 5.4).

In this implementation the **L** and **U** factors are written over the original **A** matrix and it should be noted that the leading diagonal for the **L** factor is implicitly

Algorithm 5.3: Partial pivoted Gaussian elimination

1: **for** $j = 1$ to $(N-1)$ **do** // Loop over the matrix rows
2: **for** $i = 0$ to $(j-1)$ **do** // Loop over the matrix columns
3: **if** $|a_{ii}| < threshold$ **then**
4: Find row l below i where $a_{li} = \max |a_{mi}|$ and $a_{li} \neq 0$
5: Swap rows l and i in matrix $\mathbf{A}$ and vector $\mathbf{b}$
6: **end if**
7: $F = \frac{a_{ij}}{a_{ii}}$ // Calculate the scale factor
8: **for** $k = 0$ to $(n-1)$ **do**
9: $a_{jk} = a_{jk} - (Fa_{ik})$
10: **end for**
11: $b_j = b_j - (Fb_i)$
12: **end for**
13: **end for**

Algorithm 5.4: LU decomposition

1: **for** $i = 0$ to $(N-1)$ **do** // Loop over the matrix rows
2: **for** $j = 0$ to $(i-1)$ **do** // Calculate the lower triangular matrix
3: **for** $k = 0$ to $(j-1)$ **do**
4: $a_{ij} = a_{ij} - (a_{ik}a_{kj})$
5: **end for**
6: $a_{ij} = \frac{a_{ij}}{a_{jj}}$
7: **end for**
8: **for** $j = i$ to $(N-1)$ **do** // Calculate the upper triangular matrix
9: **for** $k = 0$ to $(i-1)$ **do**
10: $a_{ij} = a_{ij} - (a_{ik}a_{kj})$
11: **end for**
12: **end for**
13: **end for**

assumed to be 1 and as a result does not need to be stored. This algorithm can suffer from numerical rounding during the calculation thus partial pivoting can be included as indicated in algorithm (5.5)

Algorithm 5.5: Partial pivoted LU decomposition

1: **for** $i = 0$ to $(N-1)$ **do** // Loop over the matrix rows
2: **if** $|a_{ii}| < threshold$ **then** // Partial pivoting
3: Find row l below i where $a_{li} = \max |a_{mi}|$ and $a_{li} \neq 0$
4: Swap rows l and i in matrix $\mathbf{A}$ and vector $\mathbf{b}$
5: **end if**
6: **for** $j = 0$ to $(i-1)$ **do** // Calculate the lower triangular matrix
7: **for** $k = 0$ to $(j-1)$ **do**
8: $a_{ij} = a_{ij} - (a_{ik}a_{kj})$
9: **end for**
10: $a_{ij} = \frac{a_{ij}}{a_{jj}}$
11: **end for**
12: **for** $j = i$ to $(N-1)$ **do** // Calculate the upper triangular matrix
13: **for** $k = 0$ to $(i-1)$ **do**
14: $a_{ij} = a_{ij} - (a_{ik}a_{kj})$
15: **end for**
16: **end for**
17: **end for**

The computational cost of calculating the LU decomposition is $\frac{2}{3}N^3$ for a non-symmetric system. If the system is symmetric then a variant of the LU decomposition can be employed called Cholesky factorisation. In the Cholesky factorisation $\mathbf{A}$ is split into factors of $\mathbf{L}$ and $\mathbf{L^T}$ thus the computation is reduced by a factor of 2 such that the computational cost is $\frac{1}{3}N^3$.

5.2 Iterative solvers

An alternative to using a direct solver is to employ an iterative solver. This process works by using a *first approximation* to the solution and then adjusting this approximation according to a particular scheme until the residual, the difference between the actual solution and the current solution, has been reduced below a certain threshold; typically 10^{-6}.

The most basic form of iterative solvers are stationary iterative solvers (Ramage, 2006). These require the matrix $\mathbf{A}$ to be split.

$$\mathbf{A} = \mathbf{M} - \mathbf{N}$$

where $\mathbf{M}$ is invertible. The linear system of equations, equation (5.1), can now be written,

$$(\mathbf{M} - \mathbf{N})\,\mathbf{x} = \mathbf{b} \tag{5.2}$$

$$\mathbf{M}\mathbf{x} = \mathbf{N}\mathbf{x} + \mathbf{b} \tag{5.3}$$

This can be formed into a sequence of iterates

$$\mathbf{M}\mathbf{x_k} = \mathbf{N}\mathbf{x_{k-1}} + \mathbf{b} \tag{5.4}$$

$$\mathbf{x_k} = \mathbf{M^{-1}}\mathbf{N}\mathbf{x_{k-1}} + \mathbf{M^{-1}}\mathbf{b}, \qquad \text{where } k = 1, 2, \ldots \tag{5.5}$$

The choice of $\mathbf{M}$ and $\mathbf{N}$ lead to different iterative methods. Common splittings include

- Jacobi

$$\mathbf{M} = \mathbf{D} \qquad \mathbf{N} = -\left(\mathbf{L} + \mathbf{L^T}\right) \tag{5.6}$$

- Gauss-Seidel

$$\mathbf{M} = \mathbf{D} + \mathbf{L} \qquad \mathbf{N} = -\mathbf{L^T} \tag{5.7}$$

where $\mathbf{D}$ is the diagonal of the matrix $\mathbf{A}$ and $\mathbf{L}$ and $\mathbf{U}$ are the strict lower and upper triangular parts of $\mathbf{A}$.

Alternatively, the solution of equation (5.1) can be written as the linear combination of products of $\mathbf{A}$ and $\mathbf{b}$ (Greenbaum, 1997), leading to the development of non-stationary iterative solvers.

$$\mathbf{A^{-1}b} \approx \hat{\mathbf{x}}(s) \in \mathcal{K}(\mathbf{A}, \mathbf{b}, s) \equiv span\left(\mathbf{b}, \mathbf{Ab}, \mathbf{A^2b}, \ldots, \mathbf{A^{s-1}b}\right) \tag{5.8}$$

where $\mathcal{K}(\mathbf{A}, \mathbf{b}, s)$ is the Krylov subspace of degree s formed by $\mathbf{A}$ and $\mathbf{b}$. Thus, the exact solution lies in $\mathcal{K}(\mathbf{A}, \mathbf{b}, n)$ but the approximate solution lies within $\mathcal{K}(\mathbf{A}, \mathbf{b}, s)$ with $s < n$. To determine the exact solution it is necessary to minimise,

$$\Phi(\mathbf{x}) = \frac{1}{2}\mathbf{x^T Ax} - \mathbf{x^T b} \tag{5.9}$$

for symmetric problems or

$$\Phi(\mathbf{x}) = (\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b}) \tag{5.10}$$

for unsymmetric systems. Solution of this minimisation problem will yield the linear combination within $\mathcal{K}$ which gives $\hat{\mathbf{x}}$.

There are a number of iterative schemes that can be applied to solve the minimisation problem and they can be split into three main categories:

- Steepest descent methods
- Conjugate gradient methods
- Generalised minimum residual (GMRES) methods

Each of these methods will be presented in turn.

5.2.1 Steepest descent methods

The method of steepest descent is one of the simplest techniques for minimising the linear function given in equation (5.9). At any point $\mathbf{x_k}$ the value of Φ decreases in the direction of negative gradient, i.e.

$$-\nabla\Phi(\mathbf{x_k}) = \mathbf{b} - \mathbf{Ax_k} = \mathbf{r_k} \tag{5.11}$$

where $\mathbf{r_k}$ is called the residual at $\mathbf{x_k}$. This is analogous to following the initial direction that a ball will move when placed on an uneven terrain; if contours are

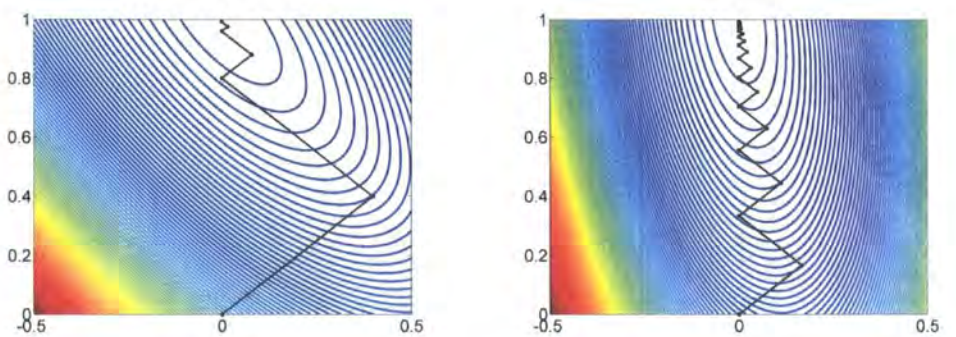
plotted on the terrain then the ball will move in the direction perpendicular to the contours.

If the residual is non-zero, then Φ can be reduced by travelling in this direction. The method of steepest descent is shown in algorithm 5.6

Algorithm 5.6: Steepest descent method

1: $\mathbf{x_0} = 0$
2: **for** $k = 0, 1, 2 \ldots$ **do**
3: $\quad \mathbf{r_k} = \mathbf{f} - \mathbf{A x_k}$
4: $\quad$ check for convergence - continue if necessary
5: $\quad \alpha_k = \frac{\mathbf{r_k^T r_k}}{\mathbf{r_k^T A r_k}}$
6: $\quad \mathbf{x_{k+1}} = \mathbf{x_k} + \alpha_k \mathbf{r_k}$
7: **end for**

The method of steepest descent can be effective but suffers if the contours of Φ are elongated as the direction of steepest descent given by $\mathbf{r_k}$. Referring to the previous analogy of a ball on uneven terrain, this situation can be considered to be a ball placed on the side of a valley and moving from one side of the valley to the other side whilst slowly moving toward the minimum at the end of the valley (figure 5.1).



(a) Problem 1 - Fast convergence (b) Problem 2 - Slow convergence

Figure 5.1: Convergence of steepest descent method (Ramage, 2006)

5.2.2 Conjugate gradient methods

The steepest descent method can be slow converging to a solution for the minimisation problem. As a result an alternative method was proposed (Hestenes and Stiefel, 1952).

The conjugate gradient method relies on the principle that the optimum search direction is not necessarily the direction of the residual $\mathbf{r_k}$. Thus, if the search direction is given by the vector $\mathbf{p_k}$ is it possible to find an optimum value for $\mathbf{p_k}$.

As the search direction is now arbitrary it must satisfy two separate requirements

- The new iterate must minimise Φ over all of the possible search directions
- We must be able to find α_k that minimises Φ in the appropriate search direction

To find an appropriate search direction $\mathbf{p_k}$ it is necessary to consider two one-dimensional steps. This situation can be represented by figure 5.2. It can be shown

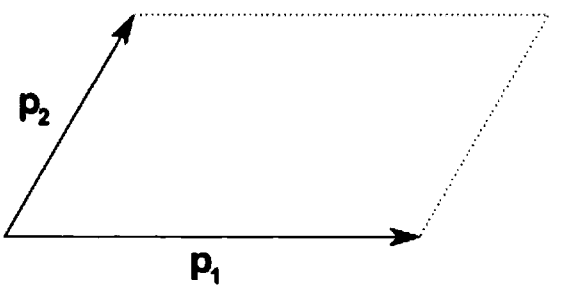


Figure 5.2: Two one-dimensional search vectors

(Kane, 1994) that to meet the two criteria specified then $\mathbf{p_1}$ and $\mathbf{p_2}$ must be **A**-*conjugate* i.e.

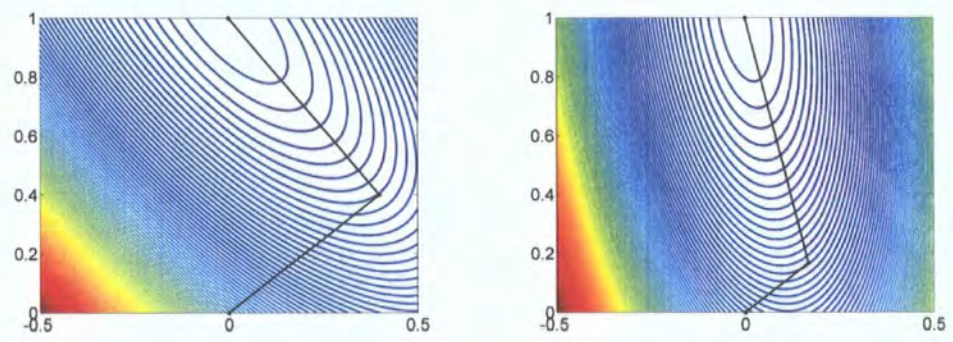
$$\mathbf{p_j^T A p_k} = 0, \qquad j < k \tag{5.12}$$

This can easily be extended to N-dimensional search vectors that are **A**-conjugate such that the N-dimensional space spanned by these vectors is minimised. This technique yields the conjugate gradient method.

In this version of the conjugate gradient algorithm there is only one matrix-vector multiplication per iteration. Figure 5.3 shows that by using conjugate search vectors

Algorithm 5.7: Preliminary conjugate gradient method

$\mathbf{x_0} = 0$; $\mathbf{r_0} = \mathbf{b}$
for $k = 1, 2 \ldots$ **do**
  check for convergence - continue if necessary
  if $k = 1$ **then**
    $\mathbf{p_1} = \mathbf{r_0}$
  else
    $\beta_k = \frac{\mathbf{r_{k-1}^T r_{k-1}}}{r_{k-2}^T r_{k-2}}$
    Let $\mathbf{p_k} = \mathbf{r_{k-1}} + \beta_k \mathbf{p_{k-1}}$
  end if
  $\alpha_k = \frac{\mathbf{r_{k-1}^T r_{k-1}}}{p_k^T A p_k}$
  $\mathbf{x_k} = \mathbf{x_{k-1}} + \alpha_k \mathbf{p_k}$
  $\mathbf{r_k} = \mathbf{r_{k-1}} - \alpha_k \mathbf{A p_k}$
end for



(a) Problem 1 (b) Problem 2

Figure 5.3: Convergence of conjugate gradient method (Ramage, 2006)

it is possible to find the solution vector in an accelerated manner when compared with steepest descent techniques. It can be shown by application of the Cayley-Hamilton theorem (Spencer, 2004) that for exact arithmetic the conjugate gradient method will converge within N iterations where N is the size of the system to be solved.

This algorithm is only valid for symmetric systems and as a result it is not possible to use it directly to solve the matrix equations that the BEM generates. This limitation is a result of the fact that for unsymmetric systems it is not possible to create orthogonal residual vectors whilst maintaining a short recurrence. To overcome this limitation a number of variants on the conjugate gradient technique have been developed to deal with unsymmetric systems.

One technique for dealing with unsymmetric systems is to form a version in which the matrix to be solved is symmetric. This can be achieved by solving the *normal equations* for the system.

$$\mathbf{A^T A x} = \tilde{\mathbf{b}} \tag{5.13}$$

where $\tilde{\mathbf{b}} = \mathbf{A^T b}$. Unfortunately this can adversely affect the rate of convergence as the condition number of the matrix to be solved now depends on the square of the original coefficient matrix.

An alternative approach employs two mutually orthogonal sequences, however this is at the cost of finding $\mathbf{x_k}$ that minimises $\| \mathbf{x_k} - \hat{\mathbf{x}} \|_A$, as a result the method does not guarantee optimal progress at each step. The residuals are provided as the augmented forms of the standard conjugate gradient residuals,

$$\mathbf{r_k} = \mathbf{r_{k-1}} - \alpha_k \mathbf{A p_k} \tag{5.14}$$

$$\tilde{\mathbf{r}}_\mathbf{k} = \tilde{\mathbf{r}}_\mathbf{k-1} - \alpha_k \mathbf{A^T} \tilde{\mathbf{p}}_\mathbf{k} \tag{5.15}$$

and the search directions are given by

$$\mathbf{p_k} = \mathbf{r_{k-1}} + \beta_k \mathbf{p_{k-1}} \tag{5.16}$$

$$\tilde{\mathbf{p}}_\mathbf{k} = \tilde{\mathbf{r}}_\mathbf{k-1} + \beta_k \tilde{\mathbf{p}}_\mathbf{k-1} \tag{5.17}$$

To ensure orthogonality we use the following relations

$$\alpha_k = \frac{\tilde{\mathbf{r}}_\mathbf{k-1}^\mathbf{T} \mathbf{r_{k-1}}}{\tilde{\mathbf{p}}_\mathbf{k}^\mathbf{T} \mathbf{A p_k}} \tag{5.18}$$

$$\beta_k = \frac{\tilde{\mathbf{r}}_\mathbf{k}^\mathbf{T}\mathbf{r}_\mathbf{k}}{\tilde{\mathbf{r}}_\mathbf{k-1}^\mathbf{T}\mathbf{r}_\mathbf{k-1}} \tag{5.19}$$

The convergence of the bi-conjugate gradient scheme can be irregular and as a result it can be difficult to compare with other iterative methods. If $\mathbf{A}$ is symmetric positive definite then the scheme results in the same answer as the standard conjugate gradient method but at twice the cost per iteration.

To overcome the problems with convergence of the bi-conjugate gradient method a stabilised version was produced by van der Vorst (1992). The stabilisation occurs by updating the residual polynomials with a linear factor, the factor is determined by solving a local steepest descent problem. The addition of this local minimisation problem causes the convergence of the bi-conjugate gradient stabilised (BiCGStab) method to be much smoother than the standard bi-conjugate gradient method. Further variants have been proposed recently to cater for specific types of problem (Gutknecht, 1993; Sleijpen and Fokkema, 1993).

5.2.3 Generalised minimum residual method

The generalised minimum residual (GMRES) method is an alternative projection based method for unsymmetric systems (Saad and Schultz, 1986).

As the matrix is unsymmetric it is not possible to employ a Lanczos method directly to ensure that the search vectors are orthogonal, thus it is necessary to employ a modified Gram-Schmidt (Schmidt, 1907) process to form the orthonormal basis for the Krylov sub-space. The modification to the Gram-Schmidt process produces Arnoldi's method (Arnoldi, 1951). In algorithm 5.8 $h_{i,k}$ is the (i,k) component of the associated Hessenberg matrix. Equation 5.20 shows the format of a Hessenberg matrix.

$$\bar{\mathbf{H}}_\mathbf{m} = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & h_{1,4} & \cdots & h_{1,m-1} & h_{1,m} \\ h_{2,1} & h_{2,2} & h_{2,3} & h_{2,4} & \cdots & h_{2,m-1} & h_{2,m} \\ 0 & h_{3,2} & h_{3,3} & h_{3,4} & \cdots & h_{3,m-1} & h_{3,m} \\ 0 & 0 & h_{4,3} & h_{4,4} & \cdots & h_{4,m-1} & h_{4,m} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{m-1,m-1} & h_{m-1,m} \\ 0 & 0 & 0 & 0 & \cdots & h_{m,m-1} & h_{m,m} \end{bmatrix} \tag{5.20}$$

Algorithm 5.8: Arnoldi's Method

1: $\mathbf{q_1} = \mathbf{r_0} \parallel \mathbf{r_0} \parallel^{-1}$
2: **for** $k = 1, 2, \ldots, m$ **do**
3: $\quad \mathbf{w_k} = \mathbf{A q_k}$
4: $\quad$ **for** $i = 1, 2, \ldots, k$ **do**
5: $\quad\quad h_{i,k} = \mathbf{q_i^T A q_k}$
6: $\quad\quad \mathbf{w_k} = \mathbf{w_k} - h_{i,k}\mathbf{q_i}$
7: $\quad$ **end for**
8: $\quad h_{k+1,k} = \parallel w_k \parallel$
9: $\quad \mathbf{q_{k+1}} = \frac{\mathbf{w}}{h_{k+1,k}}$
10: **end for**

By employing Arnoldi's method to produce the orthonormal basis the short term recurrence featured in the conjugate gradient method is lost and all of the vectors must be stored. Thus as the number of iterations increases this becomes a major drawback of the method. However, GMRES does maintain the attractive feature of the minimisation property.

The minimisation problem to be solved using GMRES is

$$\Phi(\mathbf{y}) \equiv \parallel \mathbf{b} - \mathbf{A}(\mathbf{x_0} - \mathbf{V_m y}) \parallel = \parallel \beta \mathbf{e_1} - \bar{\mathbf{H}}_\mathbf{m} \mathbf{y} \parallel \tag{5.21}$$

where $\mathbf{V_m}$ contains the m orthogonal search vectors, $\mathbf{y}$ is a vector to be minimised, $\mathbf{e_1}$ is the first column vector from an m by m identity matrix and $\bar{\mathbf{H}}_\mathbf{m}$ is a Hessenberg matrix formed in Arnoldi's method (algorithm 5.8). It can be shown (Saad, 1996) that the minimisation can be obtained from

$$\mathbf{x_m} = \mathbf{x_0} + \mathbf{V_m y_m}, \qquad \text{where} \tag{5.22}$$

$$\mathbf{y_m} = \text{argmin}_y \parallel \beta \mathbf{e_1} - \bar{\mathbf{H}}_\mathbf{m} \mathbf{y} \parallel \tag{5.23}$$

where $\text{argmin}_y f(y)$ is the value of the given argument, y, for which the the value of the expression, $f(y)$, attains its minimum value. Thus the GMRES algorithm is given by algorithm 5.9

Algorithm 5.9: GMRES

1: $\mathbf{q_1} = \mathbf{r_0} \parallel \mathbf{r_0} \parallel^{-1}$
2: **for** $k = 1, 2, \ldots, m$ **do**
3: $\quad \mathbf{w_k} = \mathbf{A q_k}$
4: $\quad$ **for** $i = 1, 2, \ldots, k$ **do**
5: $\quad\quad h_{i,k} = \mathbf{q_i^T A q_k}$
6: $\quad\quad \mathbf{w_k} = \mathbf{w_k} - h_{i,k}\mathbf{q_i}$
7: $\quad$ **end for**
8: $\quad h_{k+1,k} = \parallel w_k \parallel$
9: $\quad \mathbf{q_{k+1}} = \frac{\mathbf{w}}{h_{k+1,k}}$
10: **end for**
11: $\mathbf{y_m} = \mathbf{y}$ that minimises $\parallel \beta \mathbf{e_1} - \bar{\mathbf{H}}_\mathbf{m}\mathbf{y} \parallel$
12: $\mathbf{x_m} = \mathbf{x_0} + \mathbf{Q_m y_m}$

The only problem remaining is how to minimise the function $\parallel \beta \mathbf{e_1} - \bar{\mathbf{H}}_\mathbf{m}\mathbf{y} \parallel$. This is an overdetermined system of equations due to the size of the Hessenberg matrix. As a result it is necessary to solve it in a least squares form. An efficient method of solving this system of equations is through the use of Given's rotations to produce a **QR** factorisation of the matrix. Equation (5.24) shows an example of a Given's rotation for a 5×5 matrix.

$$\mathbf{G_{24}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & -l & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & l & 0 & m & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.24}$$

where $l = \sin\theta$ and $m = \cos\theta$. Note that this is an identity matrix with additional terms added to the (i, k) positions and as a result the matrix is orthonormal.

The Given's rotations are applied in sequence to reduce the matrix to an upper-triangular rectangular matrix. The sequence is important to ensure that no *fill-in* occurs when applying the Given's rotations. Additionally it should be noted that the matrix-matrix product is never explicitly formed and the multiplication can

be performed on individual matrix coefficients. Moreover as the algorithm iterates through k and the problem to be minimised increases in size it can be seen that the previous Given's rotations are identical, thus it is only necessary to apply the previous Given's rotations to the additional column of $\bar{\mathbf{H}}_\mathbf{m}$ and then apply the new Given's rotation to this column and the last two entries of $\beta\mathbf{e_1}$.

Finally, it can be shown that it is not necessary to explicitly calculate x_k at each iteration to check for convergence due to the relationship, (Kane, 1994),

$$|(\Phi(\mathbf{y}))|^2 = g_{k+1}g_{k+1} \tag{5.25}$$

thus the norm of the residual is the magnitude of the last term in the $\mathbf{G}\beta\mathbf{e_1}$ vector generated in the process of computing $\mathbf{y}$. This can then be compared against a convergence criterion.

As GMRES does not maintain the short recurrence property it is necessary to store all of the search vectors v_k. For small problems this is not an issue. However, for larger problems the storage requirement for all of the vectors can be an issue. As a result a restarted form of GMRES, commonly referred to as GMRES(m) (Saad and Schultz, 1986) where m is the maximum number of iterations before restarting, can be employed. By restarting the solver with the latest solution as the new initial starting vector it is possible to converge to the solution. Restarting GMRES should only be considered for large problems where storage is at a premium due to the loss of the previous storage vectors.

5.2.4 Convergence of iterative solvers

The convergence of iterative solvers is an important area of research as it is one of the determining factors that affects the overall computational cost of a particular iterative solver. As a result it is beneficial to be able to approximate how quickly a particular matrix system should converge to a solution.

Typically (Greenbaum, 1979) the condition number of a matrix has been used to estimate the rate of convergence.

$$\kappa(\mathbf{A}) = \frac{\lambda_{max}(\mathbf{A})}{\lambda_{min}(\mathbf{A})} \tag{5.26}$$

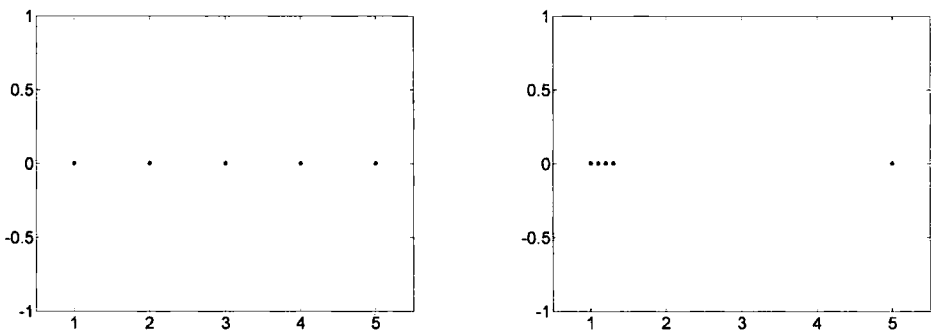
where $\kappa(\mathbf{A})$ is the condition number of the matrix $\mathbf{A}$ and $\lambda(\mathbf{A})$ is an eigenvalue of the matrix $\mathbf{A}$.

However by only using the condition number the estimate only takes account of the highest and lowest magnitude eigenvalue, and as a result a large amount of detail contained within a particular system is ignored. Consider the systems,

$$\begin{bmatrix} 2.3050 & 0.0483 & 0.1775 & 0.5269 & -0.2903 \\ 5.5023 & 5.2141 & -3.0632 & -0.2904 & -1.8667 \\ -1.1463 & 0.0261 & 3.3453 & 0.5290 & -0.0561 \\ -0.8258 & 0.0326 & 0.0805 & 4.2619 & -0.1487 \\ 7.0867 & 0.5853 & -2.8418 & 0.1354 & -0.1263 \end{bmatrix} \tag{5.27}$$

$$\begin{bmatrix} 1.1727 & 0.0352 & -0.0119 & 0.0472 & -0.0420 \\ 5.8307 & 5.2191 & -4.0186 & -0.7118 & -1.8081 \\ -0.0618 & 0.0406 & 1.1974 & 0.0461 & -0.0218 \\ -0.0202 & 0.0488 & -0.0365 & 1.3180 & -0.0343 \\ 1.3423 & 0.5142 & -0.7297 & -0.0684 & 0.6928 \end{bmatrix} \tag{5.28}$$

Both of these systems have identical condition numbers, however the distribution of eigenvalues for the systems is very different (figure 5.4). Thus, from the condition



(a) Problem 1 (equation (5.27)) (b) Problem 2 (equation (5.28))

Figure 5.4: Eigenvalue distribution for two problems with identical condition number

number alone it is not possible to determine which of these systems would converge faster.

However, if all of the eigenvalues are considered then it is possible to note that for the second problem (equation (5.28)), the majority of the eigenvalues are clustered

at one end of the eigenvalue spectrum. Greenbaum (1979) stated that the clustering of eigenvalues improves the rate of convergence. Additionally Ramage (2006) shows the rate of convergence of a selection of problems with identical condition number and matrix rank, but with intermediate eigenvalues at a variety of locations. Ramage (2006) found that the highest rate of convergence was for problems that were heavily clustered and that the slowest rate of convergence was for problems for which the eigenvalues are situated on the roots of the Chebyshev polynomial. Thus, if the eigenvalues can be caused to cluster, in particular about unity, then the rate of convergence of the iterative solver will be accelerated.

For the matrices in equations (5.27) and (5.28) it is possible to apply a GMRES solver with the following starting parameters,

$$\mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.29}$$

Figure 5.5 displays the rate of convergence for the two matrices, from this it can be seen that the effect of clustering of the eigenvalues is to accelerate the convergence of the iterative solver.

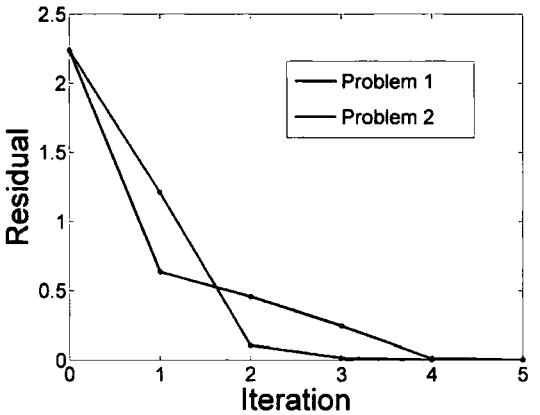


Figure 5.5: Convergence of GMRES solver for sample matrices

5.2.5 Preconditioning

We can use preconditioning to accelerate the convergence of an iterative solver. The aim of preconditioning is to alter the equation to be solved (equation (5.1)) into a simpler form which will be faster to solve.

Preconditioning can be applied in three main ways (Saad, 1996):

- Left preconditioning
- Right preconditioning
- Split preconditioning

For example left preconditioning,

$$\mathbf{MAx} = \mathbf{Mb} \tag{5.30}$$

By pre-multiplying the original matrix, $\mathbf{A}$, by a preconditioning matrix, $\mathbf{M}$, it is possible to alter the form of the new system to be solved, causing the eigenvalues of the system to be moved to more desirable locations. Consideration of left preconditioning shows that the inverse of $\mathbf{A}$ is the best preconditioner as it will result in the following matrix equation to be solved

$$\mathbf{MAx} = \mathbf{Mb} \tag{5.31}$$

$$\mathbf{A^{-1}Ax} = \mathbf{A^{-1}b} \tag{5.32}$$

$$\mathbf{Ix} = \mathbf{A^{-1}b} \tag{5.33}$$

where $\mathbf{I}$ is the identity matrix. Noting that the eigenvalues of the identity matrix are all equal to unity it can be shown that this new system can be solved in a single iteration. However, the calculation of the inverse is an $\mathcal{O}(N^3)$ operation and as such is more costly than a direct solver such as Gauss elimination. A good preconditioner therefore needs to meet a number of requirements:

- Be a good approximation to $\mathbf{A^{-1}}$
- Be computationally cheap to calculate
- Be computationally cheap to apply at each iteration

By approximating the inverse matrix the preconditioner acts to cluster the eigenvalues about unity and as a result increases the rate of convergence. These requirements are highly dependent on the matrices being solved. One type of preconditioning that meets these requirements for a particular type of equation may be of no use when dealing with an alternative problem. Thus each global case of problem needs to be studied and its characteristics extracted to allow a good preconditioner to be found.

5.2.5.1 Diagonal preconditioning

The simplest form of preconditioning that can be applied to a matrix equation is diagonal preconditioning where

$$\mathbf{M}^{-1} = \mathrm{diag}\,(\mathbf{A}) \tag{5.34}$$

As the matrices formed by application of the BEM exhibit a strong diagonal dominance (figure 5.6), then this feature should be an important part of forming the inverse. As a result using the leading diagonal as the preconditioner is a simple but potentially effective way of preconditioning the system. It can be seen that

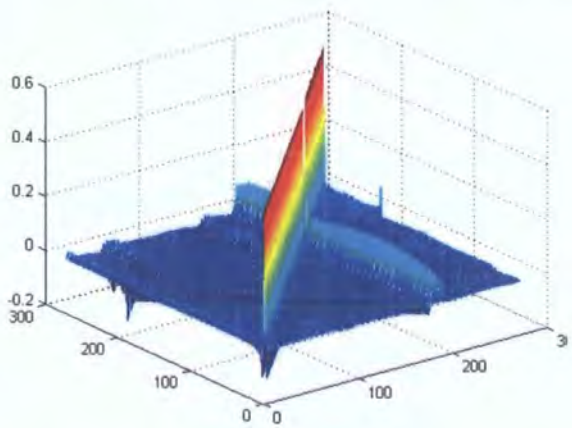


Figure 5.6: Sample matrix from the BEM showing diagonal dominance

although there is a strong diagonal dominance a large amount of detail is contained within other terms. These additional details are a result of geometric features and are affected by the application of boundary conditions (Rencis and Mann, 1997). For example, Dirichlet (displacement) boundary conditions will disturb the diagonal dominance for the corresponding degrees of freedom.

Marburg and Schneider (2003) considered the effect of diagonal preconditioning on the matrix equations generated for acoustic problems. They found that although a strong diagonal dominance was involved in the overall problem the effect of diagonal preconditioning was negligible and in certain cases increased the number of iterations to convergence when compared to no preconditioner being employed.

An expansion on the use of a diagonal preconditioner is to increase the bandwidth of the preconditioner. Popular forms of this preconditioning are tridiagonal preconditioning due to the inclusion of the terms that can be shifted off the diagonal due to boundary conditions. However, by increasing the bandwidth of the preconditioner the computational resources required for the calculation and application of the preconditioner rapidly increase and as such these are less commonly employed than a vanilla diagonal preconditioner.

5.2.5.2 Incomplete Lower-Upper preconditioning

It is possible to apply techniques similar to direct solvers such that an approximation of the direct solver result is obtained but at a reduced computational cost. Incomplete lower-upper (ILU) factorisations are an example of this technique. Instead of calculating a full LU factorisation (section 5.1.2) it is possible to calculate an approximation to this in some manner (Meijerink and van der Vorst, 1977).

The main method reducing the computational cost of both calculation of the factorisation and the application of the factorisation is to ensure that the factorisation is sparse in nature. This sparsity can be exploited from both storing the preconditioner and from employing sparse techniques for applying the preconditioner. There are two main forms of the ILU factorisation to ensure that the resulting preconditioner is sparse,

1. Sparsity pattern based.

2. Threshold based.

A sparsity pattern based ILU preconditioner employs a predefined sparsity pattern, P. This can either be a standard pattern or can be based on the current location of terms within the original $\mathbf{A}$ matrix.

Algorithm 5.10: Sparsity based ILU factorisation

1: **for** $i = 2, \ldots, N$ **do**
2: **for** $k = 1, \ldots, i-1$ and if $(i,k) \notin P$ **do**
3: $a_{ik} = \frac{a_{ik}}{a_{kk}}$
4: **for** $j = k+1, \ldots, n$ and for $(i,j) \notin P$ **do**
5: $a_{ij} = a_{ij} - a_{ik}a_{kj}$
6: **end for**
7: **end for**
8: **end for**

Algorithm 5.10 shows the format of a sparsity based ILU factorisation. From this it can be seen that the definition of the sparsity pattern P is extremely important as this will define the effectiveness of the factorisation as a preconditioner and the computational cost of the factorisation. A variant of the sparsity based ILU factorisation that reduces the emphasis of P is a sparsity based factorisation that allows *fill-in* (algorithm 5.11). These forms of the ILU are typically designated by ILU(Z) where Z is the level of fill-in that can occur during the factorisation process. The original form of the sparsity based ILU factorisation shown in algorithm 5.10 is commonly designated ILU(0). In algorithm 5.11 the term *lev* is the level of fill

Algorithm 5.11: Sparsity based ILU(Z) factorisation with fill-in

1: For all non-zero elements a_{ij} define $lev\,(a_{ij}) = 0$
2: **for** $i = 2, \ldots, N$ **do**
3: **for** $k = 1, \ldots, i-1$ and for $lev\,(a_{ik}) \leq Z$ **do**
4: $a_{ik} = \frac{a_{ik}}{a_{kk}}$
5: $a_{i*} = a_{i*} - a_{ik}a_{k*}$ // where a_{i*} denotes the i^{th} row of the matrix $\mathbf{A}$
6: Update the levels of fill, $lev_{ij} = \min\{lev_{ij}, lev_{ik} + lev_{kj} + 1\}$
7: **end for**
8: Replace any element in row i with $lev\,(a_{ij}) > Z$ by zero
9: **end for**

which is initially given by

$$lev_{ij} = \begin{cases} 0 & \text{if} \quad a_{ij} \neq 0, \text{ or } i = j \\ \infty & \text{otherwise} \end{cases} \tag{5.35}$$

This is then updated every time the element is modified in algorithm 5.11.

Application of a sparsity pattern and regulating the level of fill-in can, however, cause terms that are important to the factorisation to be dropped prematurely resulting in an increased number of iterations. As a result a threshold based ILU factorisation was presented by Saad (1994). The introduction of a thresholding criterion ensures that elements of the preconditioner that are large in magnitude will be kept within the preconditioner whereas they could be eliminated using a sparsity based ILU factorisation.

Algorithm 5.12: Threshold based ILU factorisation

1: **for** $i = 1, \ldots, N$ **do**
2: $w = a_{i*}$ // where a_{i*} denotes the i^{th} row of the matrix $\mathbf{A}$
3: **for** $k = 1, \ldots, i-1$ and when $w_k \neq 0$ **do**
4: $w_k = \frac{w_k}{a_{kk}}$
5: Apply dropping rule to w_k
6: **if** $w_k \neq 0$ **then**
7: $w = w - w_k u_{k*}$
8: **end if**
9: **end for**
10: Apply dropping rule to row w
11: $l_{ij} = w_j$ for $j = 1, \ldots, i-1$
12: $u_{ij} = w_j$ for $j = 1, \ldots, n$
13: $w = 0$
14: **end for**

A combination of the two ILU factorisation strategies can be employed to limit the amount of fill-in that can occur during the factorisation process. This involves performing a standard threshold based factorisation and then ensuring that if there

are more than Z elements in a row that only the Z terms largest in magnitude are included in the final factorisation.

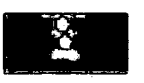
Schneider and Marburg (2003) employed a combination form of the ILU factorisation and found that it was extremely effective at reducing the iteration count associated with exterior acoustic problems. As these authors considered large scale problems it was found that the additional computation required for calculation of the ILU factorisation was negligible when compared with other costs within their analysis. Additionally, they found that convergence was not met if a preconditioner was not used.

5.3 Concluding remarks

The solution of the matrix equation given in equation (5.1) becomes a non-trivial task for situations where computational cost is at a premium. As a result it is essential to exploit the nature of the problem under consideration and to optimise the techniques employed to solve the problem.

As this thesis is concerned with small two-dimensional problems the use of a direct solver, such as an LU factorisation is considered to be reasonable. Moreover, for small two-dimensional problems iterative solvers are considered to be not optimal, however, if an iterative solver can be implemented effectively with a suitable preconditioner then this will out-perform the direct solver.

Due to the nature of the matrices created by application of the BEM it will be necessary to employ an iterative solver that is suitable for non-symmetric systems, thus the standard conjugate gradient method is not suitable. Other conjugate gradient methods could be employed. However, to deal with the non-symmetric nature of the matrices involves introducing computational mechanisms (such as matrix-vector products) which are computationally costly to implement. Thus, a GMRES solver which contains only one matrix-vector product within the iterative stage will be used. Preconditioning will be employed to accelerate convergence of the iterative solver; more details will be provided in chapter 7.



CHAPTER 6

Acceleration of the Integration Phase

The boundary element method involves the integration of fundamental solutions (previously shown in chapter 3).

Due to the relatively small size of problems under consideration within this research it has been found that the computational cost is roughly split equally between the three main parts of the analysis. Thus although for larger problems equation solution will dominate the overall computational cost, for small problems the numerical integration of the fundamental solutions equations (3.20) and (3.26) is considered to be a computationally expensive procedure. Thus, any savings that can be made in this part of the solution procedure will lead to an increase in performance allowing the rapid solution of BEM problems.

Two main techniques will be investigated to improve the speed of integration. The use of look-up tables will be introduced leading to a method which is highly effective at reducing the time required for integration but at the cost of being memory intensive and as such only suitable for higher end workstations. The second technique to be introduced will involve the fitting of equations to the integrated fundamental solutions allowing the quick computation of an approximation to the appropriate integrals. This method will be very efficient in its use of RAM and as

such suitable for a wider variety of hardware.

6.1 Look-up tables

The use of look-up tables (LUTs) to increase the rate of computation is not a new technique. Tables have been implemented for the quick extraction of trigonometric functions (NASA, 2006a,b) as well as other variables in the form of steam tables (Haywood, 1998), as illustrated in figure 6.1.

TABLE 7. SATURATED WATER AND STEAM

TEMPERATURES FROM THE TRIPLE POINT TO 100 °C

[100 kN/m² = 1 bar ≈ 14.5 lbf/in²]

Celsius temp., °C	Pressure kN/m²	Specific volume m³/kg		Specific internal energy kJ/kg		Specific enthalpy kJ/kg			Specific entropy kJ/kg K	
		Water	Steam	Water	Steam	Water	Evaporation	Steam	Water	Steam
t	p	v_f	v_g	u_f	u_g	h_f	h_{fg}	h_g	s_f	s_g
0.01	0.611	0.001000	206.2	zero	2375.6	+0.0	2501.6	2501.6	zero	9.157
2	0.705	0.001000	179.9	8.4	2378.3	8.4	2496.8	2505.2	0.031	9.105
4	0.813	0.001000	157.3	16.8	2381.1	16.8	2492.1	2508.9	0.061	9.053
6	0.935	0.001000	137.8	25.2	2383.8	25.2	2487.4	2512.6	0.091	9.001
8	1.072	0.001000	121.0	33.6	2386.6	33.6	2482.6	2516.2	0.121	8.951
10	1.227	0.001000	106.4	42.0	2389.3	42.0	2477.9	2519.9	0.151	8.902
12	1.401	0.001000	93.8	50.4	2392.1	50.4	2473.2	2523.6	0.180	8.854
14	1.597	0.001001	82.9	58.8	2394.8	58.8	2468.5	2527.2	0.210	8.806
16	1.817	0.001001	73.4	67.1	2397.6	67.1	2463.8	2530.9	0.239	8.759
18	2.062	0.001001	65.1	75.5	2400.3	75.5	2459.0	2534.5	0.268	8.713

Figure 6.1: Portion of a Steam table (from Haywood, 1998)

LUTs can be used in a number of techniques. The easiest and fastest is to use the nearest value in the table as the exact value; this is commonly referred to as a non-interpolated LUT. The second, slower, but more accurate method, is to perform some sort of interpolation between a number of points in the LUT. The use of an interpolated LUT can allow a much less refined table to be used with the same degree of accuracy as a non-interpolated LUT.

6.1.1 Displacement boundary integral equation

The boundary integral equation can be derived (equation (3.41)) as

$$c_{ij}(\kappa)\, u_i(\kappa) + \int_\Gamma T_{ij} u_j \mathrm{d}\Gamma = \int_\Gamma U_{ij} t_j \mathrm{d}\Gamma \tag{6.1}$$

where

$$U_{ij} = C_1 \left\{ C_2 \ln \left[\frac{1}{r} \right] \delta_{ij} + r_{,i} r_{,j} \right\} \tag{6.2}$$

$$T_{ij} = C_3 \left(\frac{1}{r}\right) \{r_{,n} [C_4 \delta_{ij} + 2r_{,i}r_{,j}] - C_4 [r_{,j}n_i - r_{,i}n_j]\} \tag{6.3}$$

where C_{1-4} are constants based on material properties, r is the distance between the source point and the field point and n is the unit normal.

The following analysis can be applied to both points on the boundary, by application directly to equation (6.1), or can be applied to an internal point by setting $c_{ij}(\kappa) = 1$ in equation (6.1).

We can convert the limits of the integration such that Gauss quadrature can be performed

$$u_i(\kappa) + \sum_{elem} \int_{-1}^{+1} T_{ij} N_i J(\xi) \, \mathrm{d}\xi u_i = \sum_{elem} \int_{-1}^{+1} U_{ij} N_i J(\xi) \, \mathrm{d}\xi t_i \tag{6.4}$$

where $J(\xi)$ is the Jacobian of transformation associated with the change in variable and integration limits. It can be shown that for a flat element $J(\xi) = \frac{L}{2}$, where L is the element length.

The integrals in equation (6.4) need to be computed at run-time for every source point/field point pair in the assembly of the matrices, and for every internal point/field point in the internal point solution. As a result accelerating a single integration by a small amount will accelerate the overall computation by a significant amount. It is possible to precompute the integrals from equation (6.4) and build them into a LUT such that at run-time only a few geometric parameters need to be calculated, the terms extracted from the LUTs and used directly without the need to perform a costly integration.

For a typical source point/field element pair, as shown in figure 6.2, Trevelyan and Wang (2001b) introduced four parameters to define the problem geometrically: r_m, the distance from the source point to the mid-point of the field element in question, L, the length of the element, ϕ, the angle subtended by the element to the x coordinate and θ, the angle subtended by the imaginary line of length r_m. These four parameters can be further reduced to a smaller subset of two parameters; $R_m = \frac{r_m}{L}$, a scaling parameter, and $\phi - \theta$, an angle parameter. The use of the dimensionless parameter R_m can be considered to be a scale factor of $\frac{1}{L}$ acting on the system and hence this requires the use of a modified Jacobian $\frac{J(\xi)}{L}$ in equation (6.1).

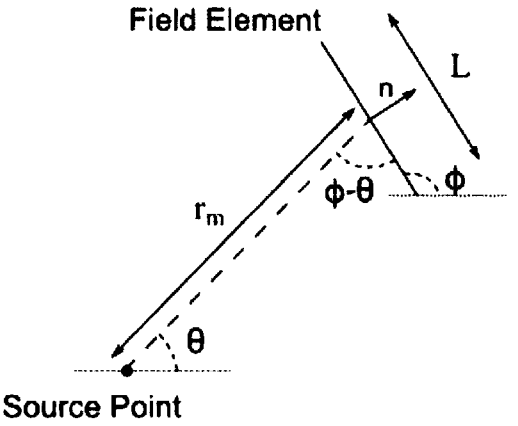


Figure 6.2: Typical source point/field element geometry (showing parameter definitions)

Consideration of the second integral from equation (6.4)

$$\int_{-1}^{+1} T_{ij} N_i J(\xi)\, \mathrm{d}\xi \tag{6.5}$$

substituting for T_{ij}

$$C_3 \int_{-1}^{+1} \left(\frac{1}{r}\right) \left\{r_{,n} \left[C_4 \delta_{ij} + 2 r_{,i} r_{,j}\right] - C_4 \left[r_{,j} n_i - r_{,i} n_j\right]\right\} N_i J(\xi)\, \mathrm{d}\xi \tag{6.6}$$

we may then create a LUT of the form

$$h_{ij}^{LUT} = C_3 \int_{-1}^{+1} \left(R_m^{-1}\right) \left\{r_{,n} \left[C_4 \delta_{ij} + 2 r_{,i} r_{,j}\right] - C_4 \left[r_{,j} n_i - r_{,i} n_j\right]\right\} N_i \frac{J(\xi)}{L} \mathrm{d}\xi \tag{6.7}$$

such that

$$h_{ij} = h_{ij}^{LUT} \tag{6.8}$$

As a result three LUTs are formed, one for each node, for each of the tensor components of T_{ij}. Consideration of the second integral from equation (6.4) a similar procedure can be followed such that we create a LUT of the form

$$g_{ij}^{LUT} = C_1 \int_{-1}^{+1} \left\{C_2 \ln\left[R_m^{-1}\right] \delta_{ij} + r_{,i} r_{,j}\right\} N_i \frac{J(\xi)}{L} \mathrm{d}\xi \tag{6.9}$$

so that

$$g_{ij} = \left[g_{ij}^{LUT} - a_{ij}\right] L \tag{6.10}$$

where

$$a_{ij} = C_1 C_2 \int_{-1}^{+1} \ln(L)\, \delta_{ij} \frac{J(\xi)}{L} N_i \mathrm{d}\xi \tag{6.11}$$

noting that L and δ_{ij} are constants and that $\frac{J(\xi)}{L} = \frac{1}{2}$ we can write

$$\begin{aligned} a_{ij} &= \frac{1}{2} C_1 C_2 \ln(L)\, \delta_{ij} \int_{-1}^{+1} N_i \mathrm{d}\xi \\ &= \mathrm{const}_{ij} \int_{-1}^{+1} N_i \mathrm{d}\xi \\ &= \alpha_{ij} \beta \end{aligned} \tag{6.12}$$

Consideration of the integral in equation (6.12) produces the well known coefficients for quadratic elements of

$$\alpha = \frac{C_1 C_2 \ln(L)\, \delta_{ij}}{12} \tag{6.13}$$

$$\beta = \begin{pmatrix} 1 & 4 & 1 \end{pmatrix} \tag{6.14}$$

By using a 2-variable LUT it is necessary to generate the LUT for a particular orientation of field element and then apply a coordinate transformation to derive the integration for particular values of ϕ and θ. The orientation of the field element in the generation of the LUT is arbitrary but has been chosen in the current work to reduce the complexity of generating the LUTs by defining the field element as being orientated with the y-axis ($\phi = 90°$), figure 6.3.

To allow the LUTs to be used for any arbitrarily angled element it is necessary to perform two separate coordinate transformations. The transformations can be represented by figure 6.4. The initial transformation applied converts a force applied to the source point in the (x, y) coordinate system into the (η, ζ) coordinate system used when generating the LUTs, as defined in figure 6.4. This is merely a rotational transformation and can be represented by equation (6.15). Notation for matrix terms is provided in the form g_{ab} where g is the appropriate sub-matrix, a is the

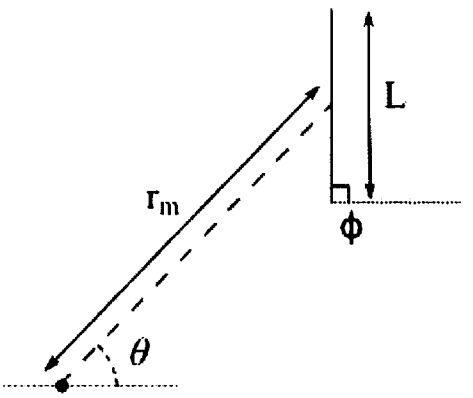
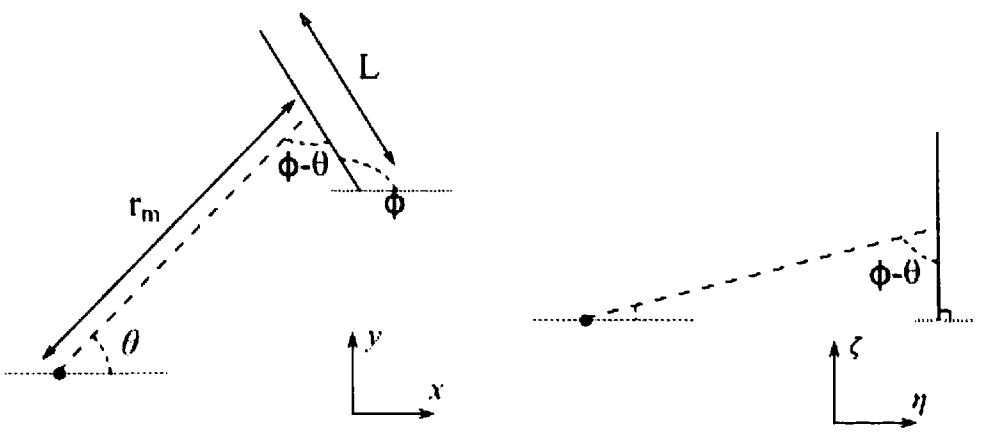


Figure 6.3: LUT integration orientation



(a) Unrotated (b) Rotated

Figure 6.4: Rotations required for arbitrary source point-field element pair

field point direction and b is the source point direction.

$$\begin{bmatrix} g_{\eta x} & g_{\eta y} \\ g_{\zeta x} & g_{\zeta y} \end{bmatrix} = \begin{bmatrix} l & m \\ -m & l \end{bmatrix} \begin{bmatrix} g_{\eta\eta} & g_{\eta\zeta} \\ g_{\zeta\eta} & g_{\zeta\zeta} \end{bmatrix} \tag{6.15}$$

where $l = \sin\phi$ and $m = \cos\phi$. The transformation gives the field displacement in terms of the (η, ζ) coordinate system after application of a force at the source point in the global (x, y) coordinate system. The resulting displacement can be transformed back to the original (x, y) coordinate system by application of a further rotational coordinate transformation.

$$\begin{bmatrix} g_{xx} & g_{xy} \\ g_{yx} & g_{yy} \end{bmatrix} = \begin{bmatrix} g_{\eta x} & g_{\eta y} \\ g_{\zeta x} & g_{\zeta y} \end{bmatrix} \begin{bmatrix} l & -m \\ m & l \end{bmatrix} \tag{6.16}$$

The transformations in equations (6.15) and (6.16) need to be applied at each node of the field element, thus we can combine the individual nodal matrix equations into a 2×6 element sub-matrix of the form

$$\begin{bmatrix} g_{xx1} & g_{xy1} & g_{xx2} & g_{xy2} & g_{xx3} & g_{xy3} \\ g_{yx1} & g_{yy1} & g_{yx2} & g_{yy2} & g_{yx3} & g_{yy3} \end{bmatrix} \tag{6.17}$$

this can be simplified to

$$\begin{bmatrix} g_{xxk} & g_{xyk} \\ g_{yxk} & g_{yyk} \end{bmatrix} \tag{6.18}$$

where k is the respective node number ($k = 1, 2, 3$). In summary equations (6.15) and (6.16) can be combined in the following matrix equation.

$$\begin{bmatrix} g_{xxk} & g_{xyk} \\ g_{yxk} & g_{yyk} \end{bmatrix} = \begin{bmatrix} l & m \\ -m & l \end{bmatrix} \begin{bmatrix} g_{\eta\eta k} & g_{\eta\zeta k} \\ g_{\zeta\eta k} & g_{\zeta\zeta k} \end{bmatrix} \begin{bmatrix} l & -m \\ m & l \end{bmatrix} \tag{6.19}$$

6.1.2 Stress boundary integral equation

Consideration of the stress boundary integral equation, equation (6.20), shows that it is of a similar overall form to the displacement boundary integral equation. For completeness the stress form of the boundary integral equation is repeated here.

$$\sigma_{ij} + \int_\Gamma S_{kij} u_k \mathrm{d}\Gamma = \int_\Gamma D_{kij} t_k \mathrm{d}\Gamma \tag{6.20}$$

where the third order tensors are

$$S_{kij} = C_5 \left(\frac{1}{r}\right)^2 \begin{bmatrix} n_i \left[2\nu r_{,j} r_{,k} + C_4 \delta_{jk}\right] + n_j \left[2\nu r_{,i} r_{,k} + C_4 \delta_{ik}\right] \\ +n_k \left[2C_4 r_{,i} r_{,j} - (1 - 4\nu)\, \delta_{ij}\right] \\ +2r_{,n} \left[C_4 \delta_{ij} r_{,k} + \nu \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j}\right) - 4 r_{,i} r_{,j} r_{,k}\right] \end{bmatrix} \tag{6.21}$$

$$D_{kij} = -C_3 \left(\frac{1}{r}\right) \left[C_4 \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j} - \delta_{ij} r_{,k}\right) + 2 r_{,i} r_{,j} r_{,k}\right] \tag{6.22}$$

Similarly to the displacement BIE, equation (6.1), we can rearrange the stress form, equation (6.20), to the following form.

$$\sigma_{ij} + \sum_{elem} \int_{-1}^{+1} S_{kij} N_i J(\xi)\, \mathrm{d}\xi u_i = \sum_{elem} \int_{-1}^{+1} D_{kij} N_i J(\xi)\, \mathrm{d}\xi t_i \tag{6.23}$$

From here we can precompute the integrals and store in memory for extraction at run-time.

$$s_{kij}^{LUT} = C_5 \int_{-1}^{+1} \left(R_m^{-1}\right)^2 \begin{bmatrix} n_i \left[2\nu r_{,j} r_{,k} + C_4 \delta_{jk}\right] + n_j \left[2\nu r_{,i} r_{,k} + C_4 \delta_{ik}\right] \\ +n_k \left[2C_4 r_{,i} r_{,j} - (1 - 4\nu)\, \delta_{ij}\right] \\ +2r_{,n} \left[C_4 \delta_{ij} r_{,k} + \nu \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j}\right) - 4 r_{,i} r_{,j} r_{,k}\right] \end{bmatrix} N_i \frac{J(\xi)}{L} \mathrm{d}\xi \tag{6.24}$$

$$d_{kij}^{LUT} = -C_3 \int_{-1}^{+1} \left(R_m^{-1}\right) \left[C_4 \left(\delta_{jk} r_{,i} + \delta_{ik} r_{,j} - \delta_{ij} r_{,k}\right) + 2 r_{,i} r_{,j} r_{,k}\right] N_i \frac{J(\xi)}{L} \mathrm{d}\xi \tag{6.25}$$

It can be seen from comparison of equations (6.22) and (6.25) that the D_{kij} term is as required but comparing equations (6.21) and (6.24) show that S_{kij} needs to be adjusted to allow for the element length.

$$s_{kij} = \frac{s_{kij}^{LUT}}{L} \tag{6.26}$$

$$d_{kij} = d_{kij}^{LUT} \tag{6.27}$$

Similarly to the displacement boundary integral equation case, coordinate transformations need to applied to the LUT value to allow it to be used for an arbitrarily oriented element. Initially we apply a transformation to relate the boundary tractions and displacements to the LUT (η, ς) coordinate system. Letting $l = \sin\phi$ and $m = \cos\phi$, we write

$$\begin{bmatrix} s_{1\eta x} & s_{2\eta x} \\ s_{1\eta y} & s_{2\eta y} \\ s_{1\varsigma y} & s_{2\varsigma y} \end{bmatrix} = \begin{bmatrix} s_{1\eta\eta} & s_{2\eta\eta} \\ s_{1\eta\varsigma} & s_{2\eta\varsigma} \\ s_{1\varsigma\varsigma} & s_{2\varsigma\varsigma} \end{bmatrix} \begin{bmatrix} l & -m \\ m & l \end{bmatrix} \tag{6.28}$$

This allows the calculation of stresses within the (η, ζ) coordinate system. The final rotation returns the stresses in the global (x, y) coordinate system by application of a rotational stress transformation (Timoshenko, 1934).

$$\begin{bmatrix} s_{1xx} & s_{2xx} \\ s_{1xy} & s_{2xy} \\ s_{1yy} & s_{2yy} \end{bmatrix} = \begin{bmatrix} l^2 & 2lm & m^2 \\ -lm & (l^2 - m^2) & lm \\ m^2 & -2lm & l^2 \end{bmatrix} \begin{bmatrix} s_{1\eta x} & s_{2\eta x} \\ s_{1\eta y} & s_{2\eta y} \\ s_{1\zeta y} & s_{2\zeta y} \end{bmatrix} \tag{6.29}$$

Combining equations (6.28) and (6.29) produces the stress boundary integral equation transformation.

$$\begin{bmatrix} s_{1xx} & s_{2xx} \\ s_{1xy} & s_{2xy} \\ s_{1yy} & s_{2yy} \end{bmatrix} = \begin{bmatrix} l^2 & 2lm & m^2 \\ -lm & (l^2 - m^2) & lm \\ m^2 & -2lm & l^2 \end{bmatrix} \begin{bmatrix} s_{1\eta\eta} & s_{2\eta\eta} \\ s_{1\eta\zeta} & s_{2\eta\zeta} \\ s_{1\zeta\zeta} & s_{2\zeta\zeta} \end{bmatrix} \begin{bmatrix} l & -m \\ m & l \end{bmatrix} \tag{6.30}$$

The transformation needs to be applied to each node of the field element and as such this can be presented in a similar manner to equation (6.19) for the displacement case

$$\begin{bmatrix} s_{1xxk} & s_{2xxk} \\ s_{1xyk} & s_{2xyk} \\ s_{1yyk} & s_{2yyk} \end{bmatrix} = \begin{bmatrix} l^2 & 2lm & m^2 \\ -lm & (l^2 - m^2) & lm \\ m^2 & -2lm & l^2 \end{bmatrix} \begin{bmatrix} s_{1\eta\eta k} & s_{2\eta\eta k} \\ s_{1\eta\zeta k} & s_{2\eta\zeta k} \\ s_{1\zeta\zeta k} & s_{2\zeta\zeta k} \end{bmatrix} \begin{bmatrix} l & -m \\ m & l \end{bmatrix} \tag{6.31}$$

where k is the node number.

This transformation is identical for the d^* term.

6.1.3 Arc elements

LUTs can be extended to cover circular arc elements. However, as the LUTs can only be used for the type of element that they were initially produced for it is important that the code that meshes the problem is optimised for this scenario.

In this work the auto-meshing code aims to use circular arc elements that subtend an angle of 30°. Figure 6.5 shows a sample element for which the meshing and remeshing code is optimised. Figure 6.6 shows parameter definitions for the circular arc element. The angle ϕ is defined as the angle between the horizontal axis and an imaginary chord line between the end-nodes of the element.

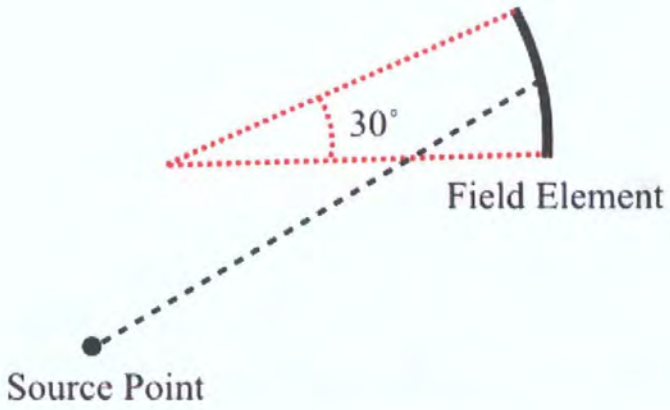


Figure 6.5: Example circular arc element

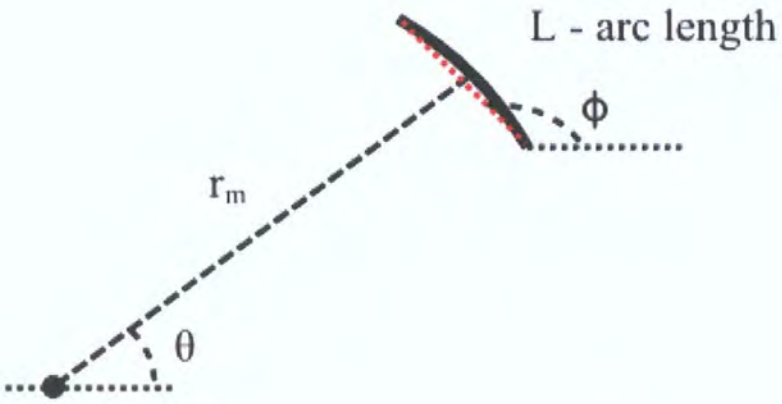


Figure 6.6: Example circular arc element with defining parameters

LUTs for this element can be generated using equations (6.9), (6.7), (6.24) and (6.25). For a circular arc element it can be shown that the Jacobian is constant with $J(\xi) = \frac{\pi R}{12}$ where R is the radius of the circular arc element. Similarly they require adjusting from the LUT form to a form suitable for use within equations (6.4) and (6.23).

Transformations are applied to allow the use of arc LUTs for arbitrary ϕ.

6.2 Refinement of LUTs

It is important to ensure that the LUTs produced using the proposed method (section 6.1) cover a suitable range of values within the scaling (R_m) direction, such that the range is wide enough that a large proportion of the integrations can be completed using the accelerated method but that the range is compact to ensure that the RAM usage is kept to a minimum.

Initial studies into the sizing of the LUTs consisted of a probabilistic study into the typical values for R_m. Figures 6.7 and 6.8 show the distribution for the integration routine when applied to the boundary and to internal points respectively.

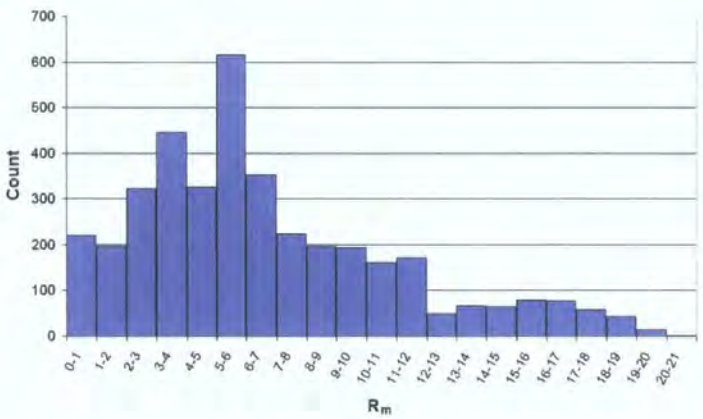


Figure 6.7: Distribution of R_m values for the boundary solution

From these figures we can see that the majority of integrations are required within a restricted band of R_m values. The maximum figure of R_m is problem dependent but by performing the study over a wide range of typical problems a useful maximum figure can be extracted. For the relatively small problems under

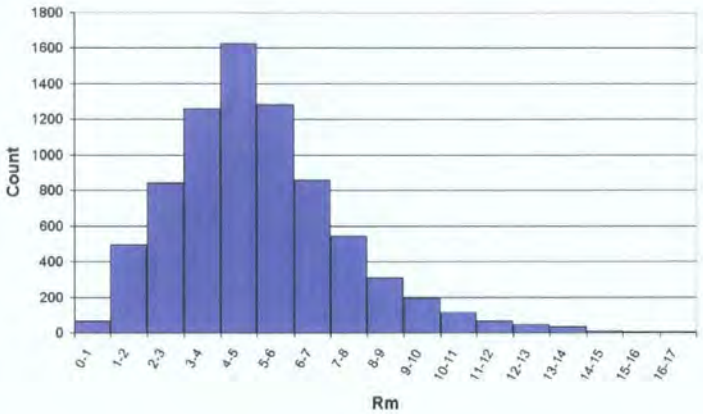


Figure 6.8: Distribution of R_m values for the internal point solution

consideration in this work a typical maximum value is $R_m = 15$.

It is also necessary to decide on a lower limit for the LUTs. Considering figures 6.7 and 6.8 it would seem sensible to extend the LUTs down to zero as values are used in this region for the singular integrals when collocating at element mid-nodes around the boundary. Difficulties arise, however, due to the singular nature of the integrals concerned and, since the non-singular integrals account for the vast majority of the cases, it was decided to use conventional integration methods for the singular integrals. It is preferable to implement a lower limit. In this work a lower limit of $R_m = 1$ has been imposed, allowing the new scheme to be used for almost all non-singular cases.

Thus, LUTs of the form described in section 6.1 are created within the following bounds

$$1 < R_m < 15$$

Integrations with values of $R_m < 1$ will be considered to be near-singular and can be treated using conventional techniques, which may be high order Gauss-Legendre quadrature or the scheme of Telles (1987). For integrals where $R_m > 15$ these may be integrated moderately economically using 2^{nd} Order Gauss-Legendre quadrature.

Consideration of the angular variable in the LUT is dependent on the type of element being employed.

6.2.1 Flat element LUTs

Figure 6.9 shows a contour display for a portion of a LUT ($h_{\eta\eta}$ for a mid-node - node 2). From this and the other plots in appendices A to D it can be seen that there are a number of lines of symmetry and anti-symmetry that can be exploited. Numerical analysis was performed upon the LUT data-set to confirm the appropriate symmetries and anti-symmetries. Lines of symmetry can be exploited at $\phi = 90°, 270°$

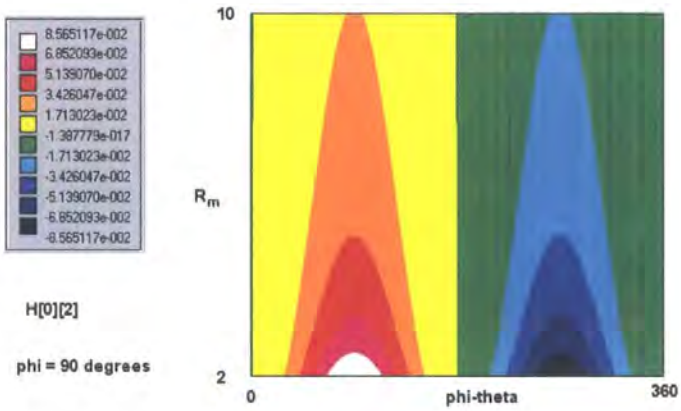
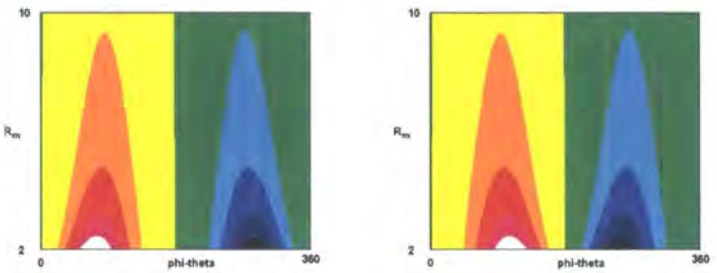


Figure 6.9: Plot of $h_{\eta\eta}^{LUT}$ for node 2

and lines of anti-symmetry at $\phi = 0°, 180°$ for all terms. Thus, for mid-nodes the entire LUT can be regenerated from a stored set over a 90° range. For end-nodes it becomes more complicated as the shape function causes the surface of the LUT to be *distorted*. Figure 6.10 shows the LUTs for nodes 1 and 3 for $h_{\eta\eta}$. This shows how the shape functions distort the surface.



(a) Node 1 (b) node 3

Figure 6.10: Plots of $h_{\eta\eta}^{LUT}$ for end-nodes

Figure 6.10 shows that although the data are distorted in the two end-nodes the plot for node 3 is a mirror image of the plot for node 1, thus we are only required to store a LUT for a single end-node. This saving can be further enhanced by noting that the distortion has only removed the lines of symmetry, but the anti-symmetry remains, and as such we are only required to store 180° of the end-node LUT.

These savings apply not only to the h^{LUT} terms but can be transferred to all of the required LUTs for both displacements and stress boundary integral equations (plots are displayed in full in appendices A, B, C and D).

Further savings can be made in storage by considering the particular case for which the LUT is generated. Equations (6.3) and (6.22) can be simplified upon knowing that the field element is aligned with the y-axis.

$$n_\eta = 1, \qquad n_\zeta = 0$$

Noting that,

$$\frac{\partial r}{\partial n} = \frac{\partial r}{\partial \eta} n_\eta + \frac{\partial r}{\partial \zeta} n_\zeta = \frac{\partial r}{\partial \eta}$$

It can be shown that

$$h_{\eta\eta} = -d_{1\eta\eta}, \qquad h_{\eta\zeta} = -d_{1\eta\zeta}, \qquad h_{\zeta\eta} = -d_{2\eta\eta}, \qquad h_{\zeta\zeta} = -d_{2\eta\zeta}$$

To store all of the necessary LUTs for flat elements requires fifteen sets of LUTs, with each set containing a mid-node LUT covering 90° and an end-node LUT containing 180°.

6.2.2 Arc element LUTs

Figure 6.11 shows a portion of an LUT ($h_{\eta\eta}$ for a mid-node - node 2). It can be seen, similarly to the case for flat elements, that there are lines of anti-symmetry that can be exploited.

Appendices E to H include plots for the g, h, s and d terms respectively.

Lines of anti-symmetry at $\phi = 0°, 180°$ can be exploited for all terms. Thus, for mid-nodes the entire LUT can be regenerated from a stored set over a 180° range. The loss of symmetry when compared with flat elements is a result of the curvature of the element. Figure 6.12 compares a flat element and a circular arc element with

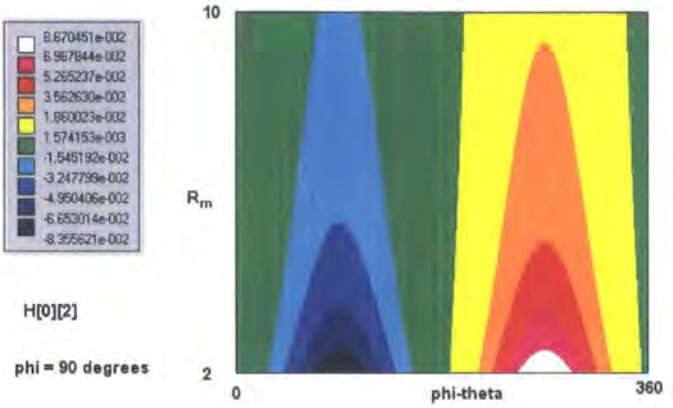
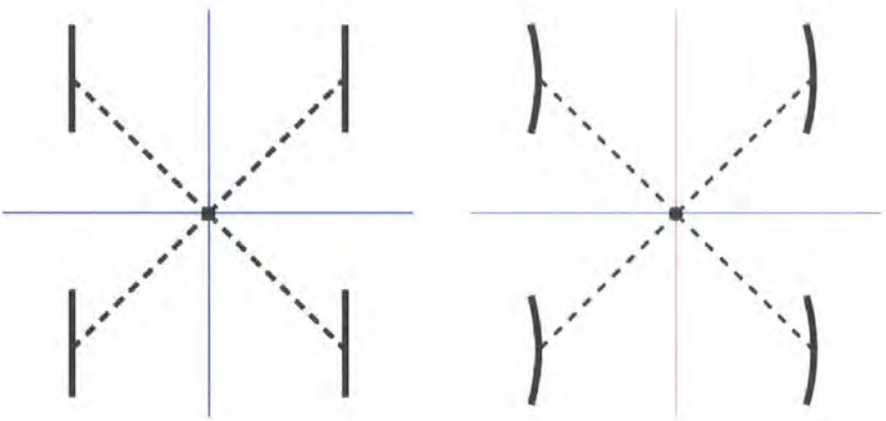


Figure 6.11: Plot of $h_{\eta\eta}^{LUT}$ for node 2 of an arc element

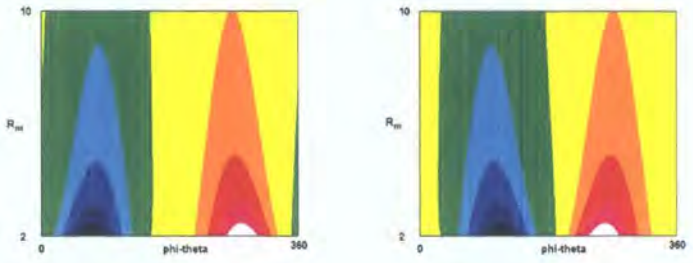
fixed ϕ as θ is increased. From this it can be seen that the flat element can be moved to any of the other four positions by mirroring along the x or y axis (designated in blue). However, consideration of a circular arc element shows that the line of symmetry along the y axis has been eliminated (designated in red). For end-nodes



(a) Flat Element (b) Circular arc element

Figure 6.12: Plots of elements for $\theta = 45°$, $135°$, $225°$ and $315°$

it becomes more complicated as the shape function causes the surface of the LUT to be *distorted*. Figure 6.13 shows the LUTs for nodes 1 and 3 for $h_{\eta\eta}$ on a circular arc element. This shows how the shape functions distort the surface. Figure 6.13 shows that although the data are distorted in the two end-nodes the plot for node 3 is a mirror image of the plot for node 1, thus in a similar manner to the flat element



(a) Node 1 (b) node 3

Figure 6.13: Plots of $h_{\eta\eta}^{LUT}$ for end-nodes of an arc element

LUTs we are only required to store a LUT for a single end-node. As a result we are required to store the complete 360° in the LUT but only for one end-node.

These savings apply not only to the h^{LUT} terms but can be transferred to all of the required LUTs for both displacements and stress boundary integral equations (plots are displayed in full in appendices E to H).

To store all of the necessary LUTs for circular arc elements requires nineteen sets of LUTs, with each set containing a mid-node LUT covering 180° and an end-node LUT covering 360°.

6.3 Error analysis of LUTs

As mentioned in chapter 1 this research is aimed at the real-time analysis for small stress analysis problems. This acceleration comes at a cost, typically, to the accuracy of the solution. For this work the maximum error has been set to 2% of the maximum principal stress, σ_{11}. This has been found to be a suitable threshold in consultation with industry.

The target error of 2% in maximum principal stress is a global error target whereas the LUT method works on a matrix term level. It is therefore necessary to find some relation between errors in the matrix terms and those in the resulting maximum principal stress. To find this relationship a test along the following lines was performed. The h and g terms for each source point/field element were calculated as usual using an adaptive Gauss-Legendre quadrature. Before inserting them

into the global matrix equation, random errors were introduced to all terms up to a prescribed maximum value. The new *randomly adjusted* matrix equations were solved and the maximum principal stress compared with a very refined mesh. Due to the random nature of the errors introduced a probablistic approach is required. Figure 6.14 displays the relationship between a known maximum introduced error on a per term basis and the resulting global error; as compared with a model composed of a very refined mesh, in maximum principal stress for a number of model sizes. The coarse model has 30 elements, standard 46 elements and the fine mesh 72 elements.

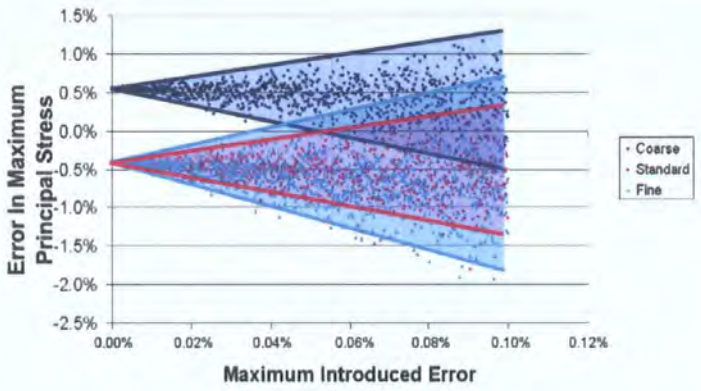


Figure 6.14: Plot of output error from known maximum introduced error

From figure 6.14 it can be seen that model size affects the way in which input error at the integration level propagates through to the output error in maximum principal stress. This difference can be attributed to the varying number of floating point operations required for each size of model, for instance the solver used (LU factorisation) requires $\frac{2N^3}{3}$ operations, thus for larger problems additional errors will be introduced due to non-exact arithmetic; faced by all computational techniques. From figure 6.14 it can be concluded that in order to achieve the stated 2% error in maximum principal stress it is necessary to have a maximum error within the integration phase of 0.1%.

Figure 6.15 shows the percentage errors associated with a coarsely defined LUT. Areas of low error, as R_m is varied, are a result of the integral parameters coinciding with a point in the LUT data-set. Additionally, the plot indicates that the

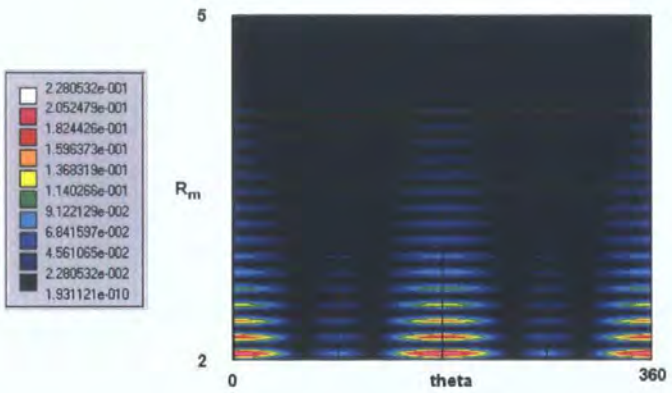


Figure 6.15: Percentage errors in coarsely generated LUT term ($g_{\eta\zeta}$ mid-node)

percentage error is higher for low values of R_m. This is expected due to the steeper gradient in the integral as $R_m \to 0$ due to the singularity. Figure 6.16 displays a surface plot for the $h^*_{\eta\zeta}$ integral for a mid-node, it can be seen that the integral becomes near-singular, displaying the associated steeper gradient, as $R_m \to 0$.

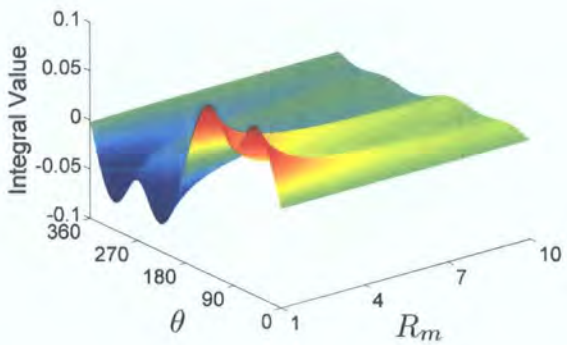


Figure 6.16: Surface plot of $h^*_{\eta\zeta}$ for a mid-node

As a result of the steeper gradient for low values of R_m a refinement scheme based on a geometric progression has been implemented of the form

$$(R_m)_i = as^{i-1} \tag{6.32}$$

where a is the initial value for R_m, noting from section 6.2 that $a = 1$, s is the geometric progression value and i is the current step number in the progression.

The use of a geometric progression causes the values in the LUT to be more closely spaced for low values of R_m, where there is a rapid gradient, but as the value of R_m increases and the gradient of the data decreases the LUT points spread.

The value of s in equation (6.32) determines the refinement and therefore the accuracy of the LUT in the R_m direction. The accuracy can be further improved by the use of interpolation within the LUT. Interpolation allows a coarser, and hence smaller, LUT to be created at additional computational cost at runtime as two values need to be extracted from the LUTs and then the interpolation performed.

6.3.1 Non-interpolated LUTs

To calculate the required value of s used in generating the LUTs, a number of LUTs were generated using a variety of values for s. These were then compared against standard Gauss-Legendre quadrature for distinct values of R_m. This was performed for a variety of values of θ. Figure 6.17 shows the relationship between R_m and the error in maximum principal stress when $s = 1.004$ as a result of the discrete nature of the LUT and the continuous nature of R_m the scatter-plot of error varies between 0 and an upper limit. Only the upper limit will be shown in further plots.

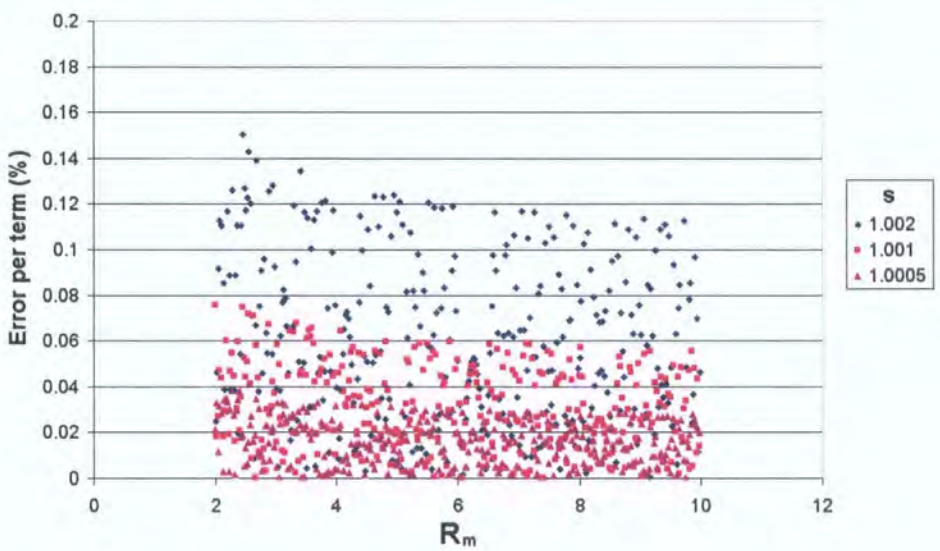


Figure 6.17: Scatter-plot of error for a non-interpolated LUT

Figure 6.18 shows the upper bounds of error in the generated plot for a non-

interpolated LUT. To meet the specification of 0.1% error in each term can be seen to require a geometric progression value of $s = 1.001$.

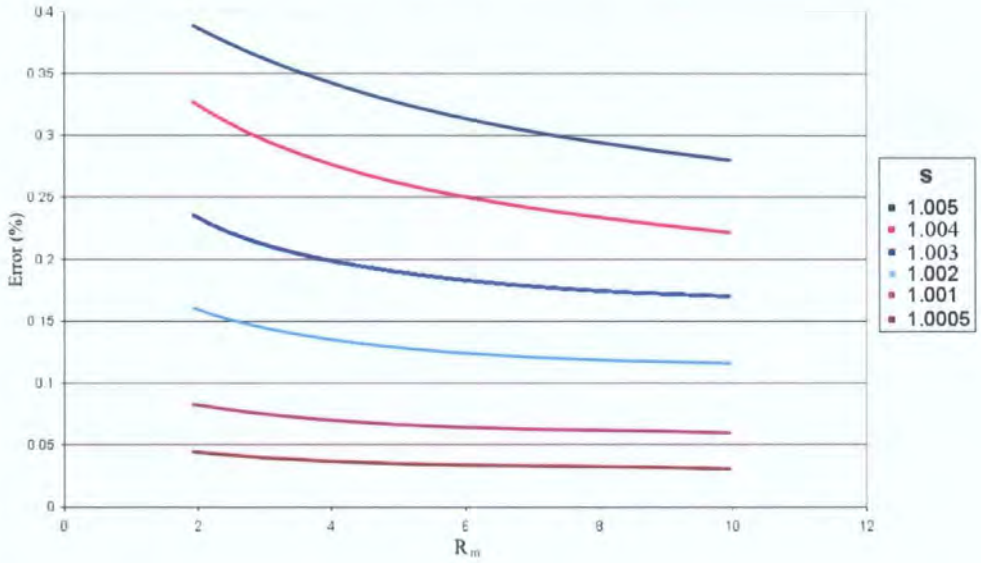


Figure 6.18: Error in maximum principal stress for non-interpolated LUT

6.3.2 Interpolated LUTs

Interpolation can be implemented in a number of manners depending on the accuracy/computational cost balance required. In this work as computational cost is a high priority the simple but relatively effective method of linear interpolation has been implemented and tested.

Figure 6.19 shows the implementation of linear interpolation. Linear interpolation requires the two values, z_{upper} and z_{lower}, in the LUT that bound our required point, z_{req}, to be extracted. The required value can then be extracted by simply plotting a straight line between the bounds and moving the required distance along this line, as

$$z_{req} \approx z' = z_{lower} + \alpha \left(z_{upper} - z_{lower} \right)$$

where

$$\alpha = \frac{x_{req} - x_{lower}}{x_{upper} - x_{lower}}$$

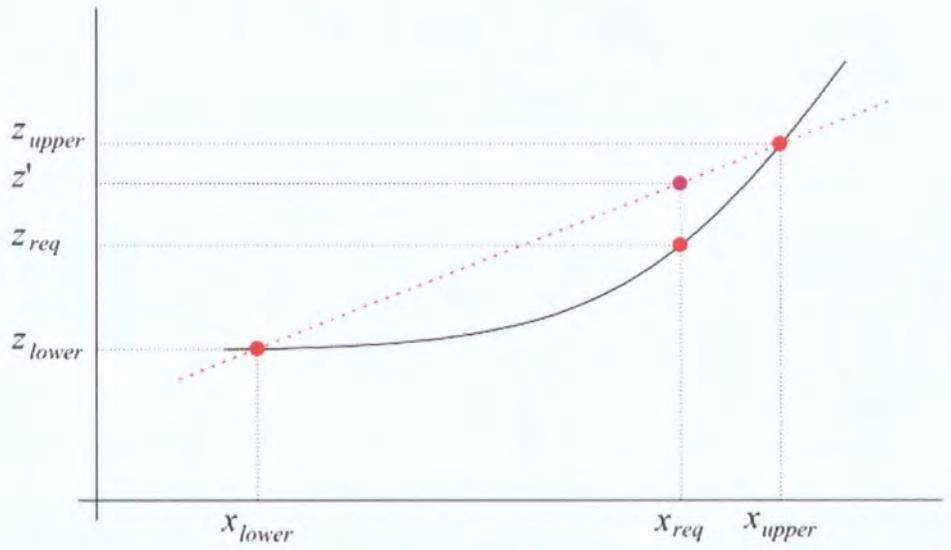


Figure 6.19: Example of linear interpolation

When interpolation was used in the LUTs the equivalent errors were distinctly lower, allowing s to be increased and thereby producing a coarser and smaller LUT. Figure 6.20 displays the upper bounds on errors for the interpolated LUT. From this it can be seen that to meet the specification of an error below 0.1% requires a geometric progression of $s = 1.03$.

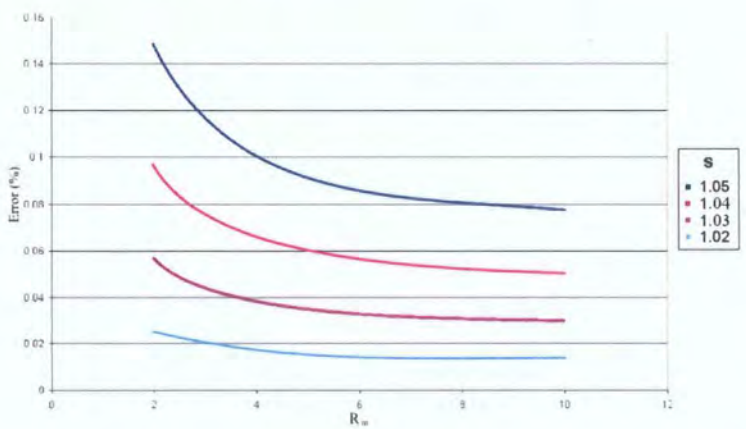


Figure 6.20: Error in maximum principal stress for interpolated LUT

6.3.3 Angular refinement

Trevelyan *et al.* (2004) concluded that a refinement in the angular parameter of 0.05° would meet the specification of an error below 0.1%.

Interpolation has not been considered within the angular parameter because of the additional computational overhead required to perform a 2-dimensional interpolation and the limited benefit offered by reducing the parameter space within this direction. This is a result of the curvature within the angular parameter being lower than the curvature within the R_m direction.

6.4 Memory requirements of LUTs

The memory requirement for the LUTs is dependent on the degree of refinement required to meet the necessary accuracy. As a result of sections 6.3.1 and 6.3.2 we require a refinement factor of $s = 1.001$ for non-interpolated LUTs and $s = 1.03$ for interpolated LUTs.

By consideration of the memory savings that can be achieved by use of symmetry and anti-symmetry, we can calculate the required amount of memory for each of the sets of LUTs. Table 6.1 displays the memory requirement for flat elements.

	Non-interpolated LUTs	Interpolated LUTs
Flat elements	838	29
Arc elements	2122	73

Table 6.1: LUT memory requirements (MB)

Table 6.1 indicates that to use non-interpolated LUTs for both flat and arc elements would require approximately 3GB of RAM. As a result it is not possible to implement non-interpolated LUTs for both flat and arc elements as a typical

personal computer would require the use of swap space* in addition to RAM to store the LUTs and operating system.

Use of interpolated LUTs for both flat and arc elements reduces the memory requirement to 102MB which can be easily accommodated by current hardware levels. However, as stated in section 6.3.2 the use of interpolated LUTs incurs additional computational cost. Therefore, it is preferable to use a mix of non-interpolated and interpolated LUTs for for the final proposed scheme.

In most simple analysis flat elements will be the predominant type of element, additionally it should be noted from table 6.1 that the memory requirement for non-interpolated flat elements is significantly lower than for arc elements resulting from the the reduced amounts of symmetry within the arc LUTs. Therefore, the proposed LUT scheme consists of using non-interpolated LUTs for flat elements and using interpolated LUTs for arc elements. Table 6.2 details the required memory for the propose scheme.

	Memory Requirement
Flat elements	838
Arc elements	73
Total	911

Table 6.2: Final LUT memory requirements (MB)

6.5 Summary of LUTs

The ability to use LUTs in place of the integrations within the boundary integral equation has been presented. An analysis of the necessary refinement has been presented allowing upper and lower bounds on the LUT to be proposed. Additionally,

*Swap space is the technique of storing data on the hard-drive instead of in RAM. This typically occurs when a program requires more memory storage than is currently available in RAM. As this technique requires access to the hard-drive instead of a memory chip it is relatively slow for access purposes.

an error analysis has been performed allowing the graduation within the LUT to be defined such that values extracted from the LUT will produce a maximum of 2% error in maximum principal stress.

Regarding the error analysis it has been noted that using current personal computer hardware it is not possible to implement the technique in an efficient manner using non-interpolated LUTs, as such a scheme has been proposed such that non-interpolated LUTs are used for the more commonly used flat elements and interpolated LUTs are used for arc elements. The proposed scheme has a total memory requirement of 911MB for the LUTs.

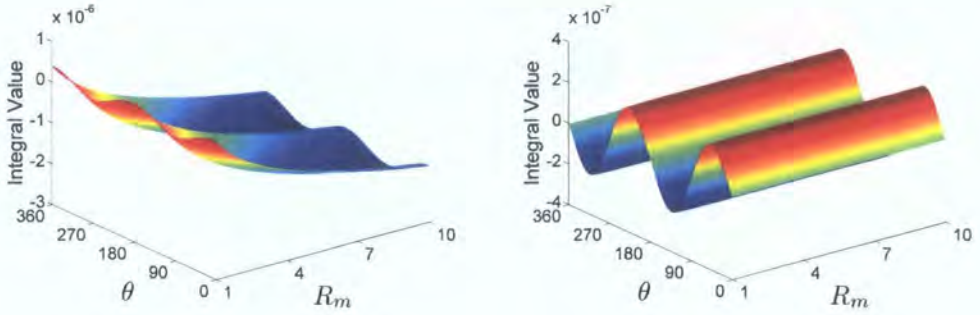
6.6 Surface fits

The use of surface fits as opposed to storing LUTs in memory and extracting values at run-time is an alternative strategy that does not suffer from the high cost in RAM that is required to store sufficiently refined LUTs. The high RAM cost limits the applicability of the LUT approach to a high specification computer.

Surface fits are similar in derivation to the LUT scheme. Whereas LUTs are generated and stored for equations (6.9) and (6.7), surface fits will be generated and the generated equations evaluated at run-time. Plots of the surfaces to be fitted are contained within appendices A to D.

Initial investigations will employ surface fits generated for the cases where $\phi = 0°$, 90°, 180° and 270°. This is a result of these cases being the most popular in the analyses that this work is aimed at. The use of a coordinate transformation to extend this technique to arbitrarily angled elements is identical to the LUT case (section 6.1).

Figure 6.21 shows two surface plots of integrals that are required for use in equation (6.4) (appendices A to D contain surface plots for all integrals). It can be seen that these are smoothly varying functions over the values of interest and as such fitting appropriate functions to the surfaces should be possible and economical.



(a) $g_{\eta\eta}$ mid-node (b) $g_{\eta\zeta}$ mid-node

Figure 6.21: Surface plots of integrals

6.6.1 Investigation of surface data

Initial research into the surface fitting of data simplified the problem to the case of fitting lines through the surface to analyse how each parameter in the integral affects the overall shape of the surface.

Figures 6.22 and 6.23 show the variation in the surface for ϕ and θ respectively for a variety of values of R_m. From these it should be noted that the variation is oscillatory in form suggesting a trigonometric basis, in the angular directions, in a surface fit expression.

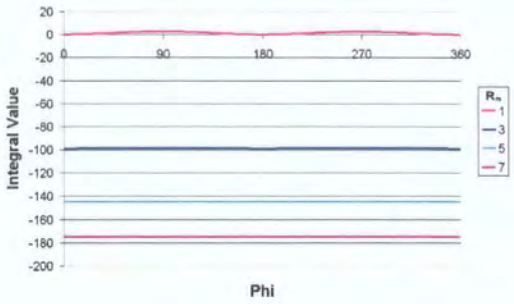
Figure 6.24 shows sections through the surface in the R_m direction. The effect of the logarithmic term can be seen in the $g^*_{\eta\eta}$ term, figure 6.24(a). As a result a basis formed from a polynomial based on a logarithmic term has been included in the basis set. However, a basis set employing logarithmic terms alone was soon found to be unsatisfactory for all terms. Hence, the basis function was expanded to include a polynomial fit in R_m^{-q}.

Application of this basis set to a selection of the integral terms confirmed that the basis set could accurately represent the data in the R_m parameter direction.

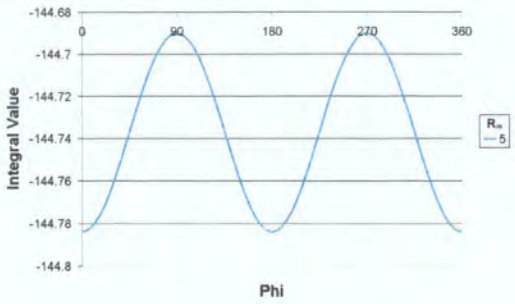
From this initial investigation it can be seen that to perform line fitting to the integral data requires the following basis functions to be used,

$$\text{Integral}_{R_m} = f\left(1, \left(\ln\left(R_m\right)\right)^q, \left(R_m\right)^{-q}\right), \qquad q = 1 \ldots 4 \tag{6.33}$$

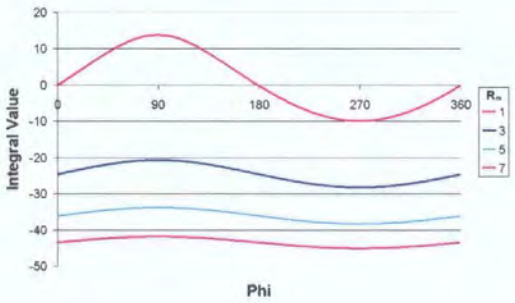
$$\text{Integral}_{\theta} = f\left(1, \sin\left(q\theta\right), \cos\left(q\theta\right)\right), \qquad q = 1 \ldots 6 \tag{6.34}$$



(a) $g_{\eta\eta}$ mid-node - macro view

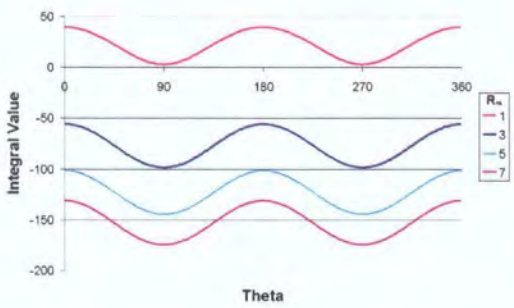


(b) $g_{\eta\eta}$ mid-node - zoomed view

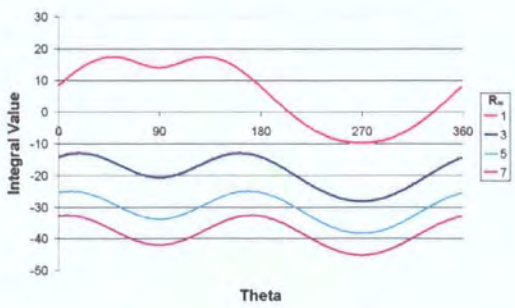


(c) $g_{\eta\eta}$ end-node

Figure 6.22: Line plots for variable ϕ, $\theta = 90°$

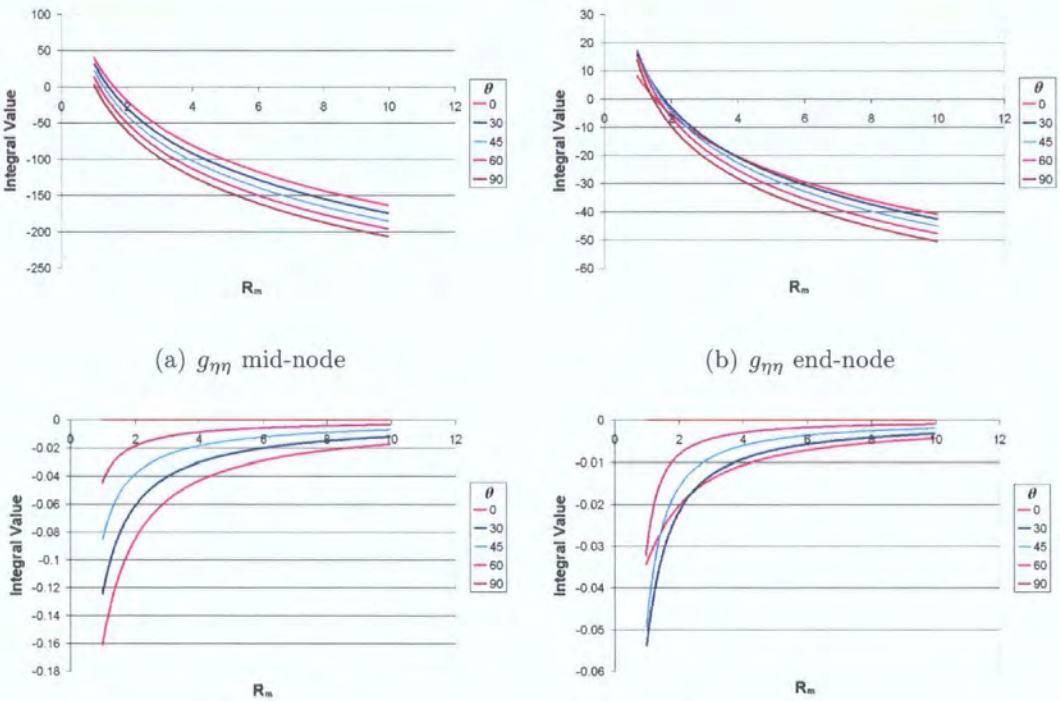


(a) $g_{\eta\eta}$ mid-node



(b) $g_{\eta\eta}$ end-node

Figure 6.23: Line plots for variable θ, $\phi = 90°$



(a) $g_{\eta\eta}$ mid-node

(b) $g_{\eta\eta}$ end-node

(c) $h_{\eta\eta}$ mid-node

(d) $h_{\eta\eta}$ end-node

Figure 6.24: Line plots for variable R_m, $\phi = 90°$

To extend the work to a 2-parameter surface fit a least squares fit can be performed using a basis set formed from the basis sets listed in equations (6.33) and (6.34).

$$I = \left\{ f = W(\theta) X(R_m) : \begin{array}{l} W \in (1, \sin(j\theta), \cos(j\theta)), j = 1, \ldots, 6 \\ X \in \left(1, [\ln R_m]^k, R_m^{-k}\right), k = 1, \ldots, 4 \end{array} \right\} \quad (6.35)$$

This results in 117 surface basis functions. For any individual matrix term, a significant proportion of the surface basis functions will have a coefficient close to zero, allowing the corresponding basis functions to be removed from the surface fit without reducing the error of the fit significantly. The problem then becomes the least squares fitting of surfaces of the form of figure 6.21 such that the computational effort to evaluate the resulting expression is minimised, within a constraint of a 0.1% upper bound on error.

6.6.2 Surface fit methodology

To achieve the optimum least squares surface fit a number of stages to the fitting procedure have been implemented. The first stage of surface fitting is a progressive reduction scheme as outlined in algorithm 6.1,

Algorithm 6.1: Progressive reduction scheme for surface fitting

1: Initialise $n = 117$
2: **while** $n > 1$ **do**
3: Perform least squares fit using n basis functions
4: Determine the *importance* of each basis function by weighting the L_2 norm of the basis function by the least squares coefficient
5: Reject the least important basis function
6: $n := n - 1$
7: **end while**

Application of algorithm 6.1 produces a list of basis functions in order of importance, this being defined as the L_2 norm of the basis function over the area of interest multiplied by the corresponding least squares coefficient. This takes into account

the difference in magnitude between the various basis functions. For instance, R_m^{-1} could be orders of magnitude different to R_m^{-4} and as such the importance function should allow for this. For example, if $R_m^{-1} = 0.1$ then $R_m^{-4} = 10^{-4}$. The relative importance of these two basis functions to the surface fit must include the magnitude of the terms and not only the surface fit coefficients.

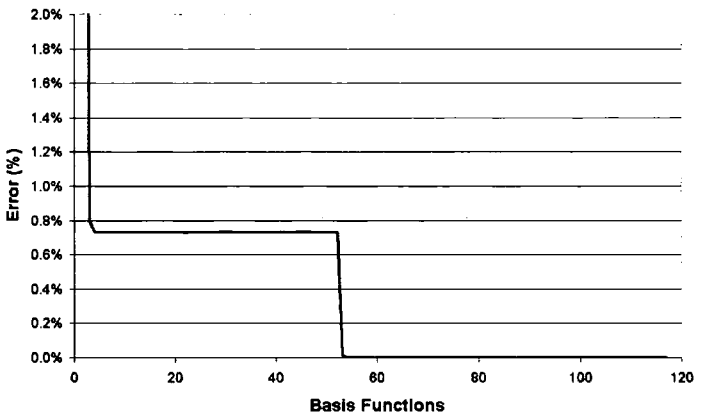


Figure 6.25: Typical plot of error as the number of basis functions is reduced

Figure 6.25 shows how the percentage error in a typical surface fit varies as the number of basis functions is reduced. It can be seen that, although the *least important* function is being eliminated each iteration, there are step changes in the error level. These step changes imply that a function that is important to the final surface fit has been eliminated from the set of basis functions. Thus, a second stage of surface fitting is required to achieve the final fit.

Algorithm 6.2 will produce the optimum surface fit from the functions passed from the progressive reduction algorithm . This might of course be applied directly to the original basis functions. However, due to the number of basis functions involved this brute force approach is not feasible due to the high computational cost incurred.

Algorithm 6.1 has been combined with algorithm 6.2 to automate the process of performing a least squares surface fit. Extraction of the important basis functions from the progressive reduction scheme is achieved by consideration of the difference in error between successive analysis, if the difference in error is greater than a prescribed level; 10^{-7} was found to be a suitable level for this work, then the basis

Algorithm 6.2: Secondary surface fitting stage

```
Extract Important Basis functions from progressive reduction scheme
while ε < 0.1% do
    for all Basis functions do
        Calculate least squares surface fit with current basis function eliminated
        Calculate error norm ε_i
    end for
    Find surface fit, k, with lowest error norm
    ε = ε_k
    Eliminate basis function with index k
end while
Return final equation set
```

function is noted and passed into algorithm 6.2. A low tolerance has been implemented at this stage as the brute force approach of algorithm 6.2 will eliminate all of the unnecessary terms from the surface fit.

6.6.3 Surface fit equations

Application of the algorithms in section 6.6.2 produces equations for each node for each of the fundamental solutions. The algorithm is applied to data with values of $\phi = 0°$, $90°$, $180°$ and $270°$.

Additionally, the data at low values of R_m vary rapidly, as illustrated, for example, in figure 6.24, and as a result the dataset is split into two sections and a least squares fit applied to the two sub-datasets.

- $2 < R_m \leq 3$
- $3 < R_m \leq 15$

Appendices I to L contain all of the 408 surface fit equations generated by the algorithms presented.

Restricting the current analysis to $\phi = 90°$ for $3 < R_m \leq 15$, a set of surface fit

expressions are presented. The simplest expressions are for the mid-node elements.

$$g_{002} = [2.166 + 2.158\cos(2\theta) - 8.996\ln R_m] \times 10^{-7} \tag{6.36}$$

$$g_{012} = g_{102} = \left[2.166\sin(2\theta) - 0.108R_m^{-2}(\sin(2\theta) + \sin(4\theta))\right] \times 10^{-7} \tag{6.37}$$

$$g_{112} = [2.166(1 - \cos(2\theta)) - 8.996\ln R_m] \times 10^{-7} \tag{6.38}$$

where g_{ijk} is the expression for the term with source point direction i, field element direction j for node k where $k = 1, 2, 3$. Corresponding expressions are formed for the end-nodes. As a result of the shape functions the dataset is distorted and hence these expressions are more complicated. However, savings can be made by noting that all of the end-node expressions can be represented in the following format.

$$g_{001} = A + B \tag{6.39}$$

$$g_{003} = A - B \tag{6.40}$$

where

$$A = \left[0.541(1 + \cos(2\theta)) - 2.249\ln R_m - 0.25R_m^{-2}\cos(2\theta)\right] \times 10^{-7} \tag{6.41}$$

$$B = \left[R_m^{-1}(1.395\sin(\theta) + 0.266\sin(3\theta))\right] \times 10^{-7} \tag{6.42}$$

Similar savings can be made for the traction kernel, for example

$$h_{001} = C + D \tag{6.43}$$

$$h_{003} = C - D \tag{6.44}$$

where

$$C = \begin{bmatrix} -R_m^{-1}(0.03515\cos(\theta) + 0.00862\cos(3\theta)) \\ +R_m^{-3}(0.00786\cos(3\theta) + 0.00373\cos(5\theta)) \end{bmatrix} \tag{6.45}$$

$$D = \begin{bmatrix} -R_m^{-2}(0.02188\sin(2\theta) + 0.00862\sin(4\theta)) \\ +R_m^{-4}(0.00458\sin(4\theta) + 0.00250\sin(6\theta)) \end{bmatrix} \tag{6.46}$$

In order to accelerate further the evaluation of these expressions, it is effective to build a look-up table containing values of $\sin(q\theta)$, $\cos(q\theta)$ and $\ln(R_m^q)$ for various arguments, thereby avoiding the lengthy computation times associated with the evaluation of trigonometric and logarithmic functions.

An arbitrary element orientation, ϕ, can be considered by applying a coordinate transformation in exactly the same way as was described for the LUT approach. However, the author finds it effective to use surface fit expressions directly for the common cases $\phi = 0°$, $90°$, $180°$ and $270°$ since the coordinate transformations involve extra computational cost.

Equations (6.36) to (6.46), and the equivalent surface fit expressions for the other terms included within appendices I to L, may be computed very rapidly and the results placed directly into the boundary element matrices as a complete replacement for conventional numerical integration. Apart from the coding of the surface fit expressions and the small LUTs for trigonometric and logarithmic expressions, this approach incurs no memory cost and in this respect is advantageous over the LUT approach

6.7 Concluding remarks

In this chapter two separate techniques have been proposed to accelerate the computation of the integral terms required for use within the boundary element method for elastostatics.

The methods are derived from a similar basis in that the integrals can be precomputed in some manner and then stored for later retrieval at a low computational cost. As a result it is possible to use a high order Gauss-Legendre scheme to calculate the original dataset. Thus, storage of this high order data-set will produce a higher accuracy LUT or surface fit when compared with an adaptive scheme, this allows a potentially lower refinement in LUT or a reduced number of terms in the surface fits.

The first technique involves the storage of the integral data within LUTs stored in memory. Refinement of the LUTs has been discussed to ensure that errors are kept within 0.1% of the maximum principal stress. Symmetries within the angular direction in the LUT have been exploited to reduce the amount of data that is required to be stored. Additionally, terms that repeat due to the orientation of the field element have been stored only once. The use of interpolation within the R_m

parameter direction has been investigated and two refinement levels suggested; one for interpolated LUTs and one for non-interpolated LUTs. Moreover, the memory requirement for the LUTs has been presented as this has a major influence on the viability of LUTs within the engineering sector.

The second technique proposed involves performing a least squares surface fit to the dataset and then computing these comparatively simple equations instead of performing Gauss-Legendre quadrature. As the dataset varies rapidly within the R_m direction it has been split into multiple intervals and separate least squares fits performed across these intervals. This has reduced the number of terms required to accurately describe sections of the dataset. Moreover, the surface fits have been found to have symmetries within the end-nodes allowing the computation of both end-nodes at once reducing the overall computational cost.

CHAPTER 7

Acceleration of the Solution Phase

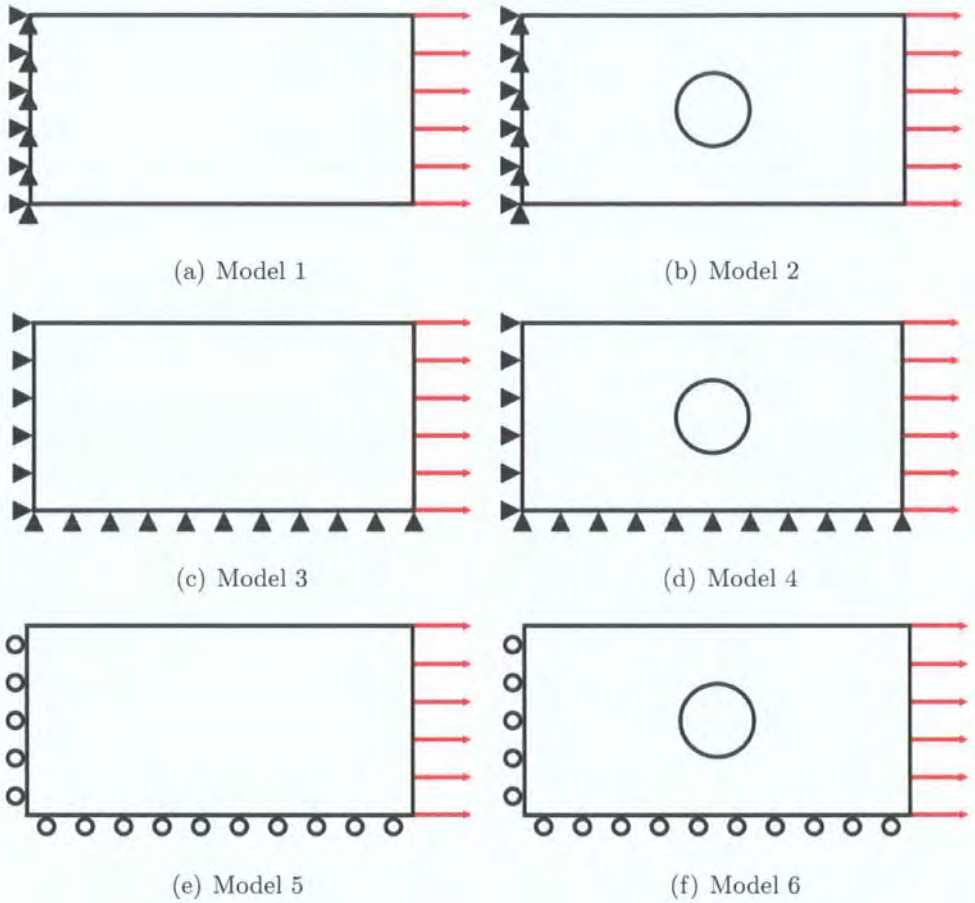
The application of the boundary element method (BEM) results in a system of equations that can be formulated in the following matrix problem,

$$\mathbf{Ax} = \mathbf{b} \tag{7.1}$$

As a result it is necessary to solve this problem in an efficient manner. Chapter 5 has introduced current techniques that allow the rapid solution of matrices using both direct and iterative solvers. Of this previous work the use of iterative solvers in combination with a suitable preconditioner offers the biggest opportunity for reducing the computational cost in the matrix solution phase. However, as the type of problems under consideration are small two-dimensional problems, in particular under reanalysis, the use of a direct solver, if it can be beneficial for future analysis, will be considered.

In this chapter consideration will be given to suitable preconditioners that can be applied to the matrices generated by the BEM. The advantages and disadvantages of each method will then be considered and a new form of preconditioner suitable for small reanalysis problems will be proposed. This will be combined into an overall strategy for the solution of matrices under reanalysis conditions.

In this chapter six models have been used to for comparison purposes; figure 7.1 shows the geometries of the six models.



(a) Model 1 (b) Model 2

(c) Model 3 (d) Model 4

(e) Model 5 (f) Model 6

Figure 7.1: Models used within equation solution analysis

These models cover two simplistic geometries popular in current literature and feature three different applications of boundary condition so that the effect of boundary conditions can be considered. Models 3 and 5 as well as models 4 and 6 are physically identical, however, they are implemented numerically different as a result of the corresponding boundary conditions.

7.1 Initial scheme

The Concept Analyst software (Trevelyan, 2003) used as a background for this research is aimed at problems in which reanalysis is likely to occur. As a result a number of techniques have already been implemented in the software to accelerate

the computation and solution of the BEM. Details of these methods can be found in Trevelyan and Wang (2001a) and Trevelyan and Wang (2001b).

The procedures currently implemented within the Concept Analyst software will be used as a baseline to establish the effectiveness of proposed techniques. The Concept Analyst framework for reanalysis can be represented by the flowchart in figure 7.2. The implementation of a direct solver for the initial solution is considered

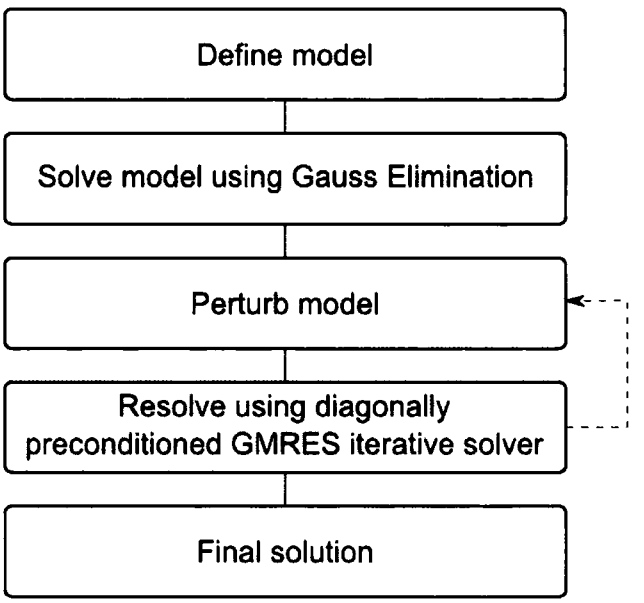


Figure 7.2: Current Concept Analyst solution framework

to be a reasonable assumption as the user may want to take one of a number of possible routes within the Concept Analyst software, for instance,

- Show a deformed plot of the solution.
- Display contours of displacement.
- Display contours of a particular stress.

As a result, the additional time required to complete a direct solve of the initial problem is considered insignificant when compared to the time required to decide upon the next course of action and implement it.

If the geometry is modified when displaying a contour plot the problem becomes one of reanalysis. It is possible to use the previous solution as an initial estimate for

the current problem in an iterative solver. GMRES is the iterative solver employed since it can accommodate unsymmetric matrices. As this research is concerned with problems of reanalysis, and as a result iteration counts will be relatively low, the loss of the short term recurrence within the orthogonal vectors is not a drawback for GMRES. To accelerate the rate of convergence of the iterative solver it is necessary to use a preconditioner within the iterative solver. Due to the ease of calculation and application a diagonal preconditioner is currently used.

7.2 Improving matrix condition

The application of particular boundary conditions, in particular normal displacement boundary conditions (models 5 and 6), can cause the form of the **A** matrix to deteriorate as the eigenvalues separate, thus reducing the, potentially, clustered nature. This is a result of a 2×2 sub-matrix on the leading diagonal being rotated such that the dominant terms are moved off of the leading diagonal and replaced with smaller terms.

$$\begin{bmatrix} 0.0000 & 0.5000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.0058 & 0.0000 & -0.0223 & 0.0722 & -0.0699 & 0.0392 \\ 0.0000 & 0.0000 & 0.0000 & 0.5000 & 0.0000 & 0.0000 \\ -0.0440 & -0.1114 & 0.0100 & 0.0000 & -0.0440 & 0.1114 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.5000 \\ -0.0699 & -0.0392 & -0.0223 & -0.0722 & -0.0058 & 0.0000 \end{bmatrix} \tag{7.2}$$

Equation (7.2) displays a 6×6 sub-matrix situated on the leading diagonal for a problem involving normal displacement boundary conditions. The terms in red are situated on the leading diagonal of the original matrix. However, the terms in blue are larger and as a result it would be beneficial for a matrix solver if these terms were moved to the leading diagonal. The rotation of the sub-matrices on the leading diagonal causes the eigenvalues of the system to spread out and as a result the rate of convergence of an iterative solver will be reduced.

To improve the rate of convergence it is possible to implement a row swapping strategy in which rows relating to the same collocation point are exchanged if such an

exchange leads to the larger terms moving to the leading diagonal. It is necessary to

Algorithm 7.1: Row-swapping strategy

1: **if** Model contains normal displacement boundary conditions **then**
2: **for** $i = 0, \ldots, \frac{N}{2} - 1$ **do** // where N is the number of rows.
3: **if** $|a_{2i+1,2i+1}| + |a_{2i+2,2i+2}| < |a_{2i+1,2i+2}| + |a_{2i+2,2i+1}|$ **then**
4: Swap rows and store row numbers
5: **end if**
6: **end for**
7: **end if**

store the row numbers that are exchanged so that the same rows can be returned to the original locations before performing a subsequent reanalysis as this will assume conventional ordering to the **A** matrix.

$$\begin{bmatrix} -0.0058 & 0.0000 & -0.0223 & 0.0722 & -0.0699 & 0.0392 \\ 0.0000 & 0.5000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.0440 & -0.1114 & 0.0100 & 0.0000 & -0.0440 & 0.1114 \\ 0.0000 & 0.0000 & 0.0000 & 0.5000 & 0.0000 & 0.0000 \\ -0.0699 & -0.0392 & -0.0223 & -0.0722 & -0.0058 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.5000 \end{bmatrix} \tag{7.3}$$

Application of algorithm 7.1 causes the terms with high magnitudes to be returned to the diagonal locations causing the condition number of the matrix, and hence the overall clustering of the matrix, to be improved. Equation (7.3) shows the effect of algorithm 7.1 on the matrix shown in equation (7.2).

7.3 Preconditioning

To improve the rate of convergence for the iterative solver it is necessary to implement a different preconditioner that is more suitable for the matrix problem to be solved than the currently implemented diagonal preconditioning. Diagonal preconditioning is popular due to the simplicity of calculating and applying the precondi-

tioner. However, as the preconditioner is merely the leading diagonal of the matrix, there is a limited amount of information carried into the preconditioner. This has been noted by Marburg and Schneider (2003) for acoustic BEM problems in which diagonal preconditioning actually *increased* the iteration count for the employed solver.

An alternative to the diagonal preconditioner that includes information off of the matrix diagonal and as a result carries more information into the preconditioner is the incomplete lower-upper (ILU) factorisation. This can be applied to an iterative solver as a split preconditioner as it has two components.

$$\mathbf{L}^{-1}\mathbf{A}\mathbf{U}^{-1}\mathbf{z} = \mathbf{A}^{-1}\mathbf{b}, \qquad \mathbf{x} = \mathbf{U}^{-1}\mathbf{z} \tag{7.4}$$

The advantage of applying the preconditioner in the split form is that both the **L** and **U** factors can be applied via a forward or back-substitution and do not require a matrix-matrix product.

Due to the dense nature of the matrix equations generated by the BEM, it would be impractical to employ a sparsity pattern based on the current layout of the matrix as it would become a full LU decomposition. As a result a threshold based ILU factorisation has been implemented within the reanalysis code. The threshold technique employs a sparse LU factorisation as the basis, however, upon calculation of the appropriate l and u terms they are checked against a threshold such that,

$$l_{i,j} = \begin{cases} \text{Calculated } l_{i,j} & if \quad l_{i,j} > \text{threshold} \\ 0 & \text{otherwise} \end{cases} \tag{7.5}$$

An identical drop rule is applied to the u terms. Algorithm 7.2 has been applied to a number of models with a selection of perturbations. Moreover, the threshold has been varied to determine if there is a generally acceptable *sweet-spot* of threshold for the types of problems under consideration in this thesis. Variation in the threshold results in variation in the number of terms that are dropped using the strategy. The number of terms dropped can be normalised with respect to the size of the problem concerned; this is defined as the sparsity of a matrix and is given by,

$$\text{Sparsity} = \frac{\text{Number of terms not dropped}}{\text{Total number of terms}} \tag{7.6}$$

Algorithm 7.2: ILU threshold based GMRES scheme

1: **for** $i = 1, \ldots, N$ **do**
2: $w = a_{i*}$ // where a_{i*} denotes the i^{th} row of the matrix $\mathbf{A}$
3: **for** $k = 1, \ldots, i-1$ and when $w_k \neq 0$ **do**
4: $w_k = \frac{w_k}{a_{kk}}$
5: Apply dropping rule to w_k
6: **if** $w_k \neq 0$ **then**
7: $w = w - w_k u_{k*}$
8: **end if**
9: **end for**
10: Apply dropping rule to row w
11: $l_{ij} = w_j$ for $j = 1, \ldots, i-1$
12: $u_{ij} = w_j$ for $j = 1, \ldots, n$
13: $w = 0$
14: **end for**
15: $\mathbf{q_1} = \mathbf{r_0} \parallel \mathbf{r_0} \parallel^{-1}$
16: **for** $k = 1, 2, \ldots, m$ **do**
17: $\mathbf{w_k} = \mathbf{A}\mathbf{L}^{-1}\mathbf{q_k}\mathbf{U}^{-1}$
18: **for** $i = 1, 2, \ldots, k$ **do**
19: $h_{i,k} = \mathbf{q_i^T w_k}$
20: $\mathbf{w_k} = \mathbf{w_k} - h_{i,k}\mathbf{q_i}$
21: **end for**
22: $h_{k+1,k} = \parallel w_k \parallel$
23: $\mathbf{q_{k+1}} = \frac{\mathbf{w}}{h_{k+1,k}}$
24: **end for**
25: $\mathbf{y_m} = \mathbf{y}$ that minimises $\parallel \beta\mathbf{e_1} - \bar{\mathbf{H}}_\mathbf{m}\mathbf{y} \parallel$
26: $\mathbf{x_m} = \mathbf{x_0} + \mathbf{L}^{-1}\mathbf{Q_m}\mathbf{y_m}\mathbf{U}^{-1}$

Due to the large variation in the threshold level it is typically plotted on a logarithmic scale. Figure 7.3 shows the classical S shape commonly seen in drop strategies. This

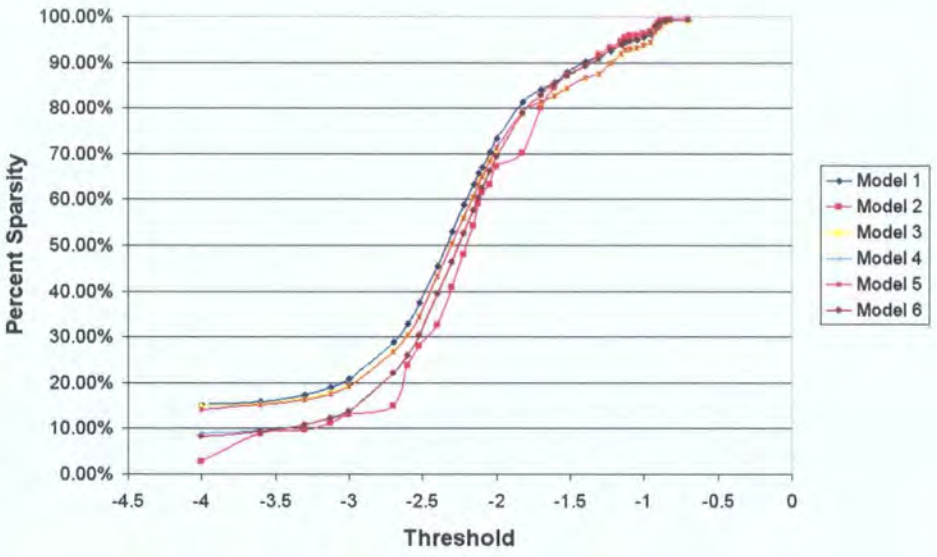


Figure 7.3: Sparsity of ILUT preconditioner

is a result of the distribution of terms within the matrix. It should be noted that to avoid the ILU factorisation producing a singular preconditioner it is forced to include the diagonal terms even if they are smaller in magnitude than the threshold. Table 7.1 shows the variation of iteration count with threshold for two problems, where N is the size of the $\mathbf{A}$ matrix. As the threshold is applied to the decomposed system and not the original $\mathbf{A}$ it is necessary to have an upper threshold of 40 as this results in a diagonal preconditioner. For thresholds between 0.1 and 30 (for model 1) the effect of non-exact arithmetic can be seen as the GMRES fails to converge to a solution within the required number of iterations as expected by Cayley-Hamilton theorem. This is a result of the *orthogonal* vectors created in the iterative process not being orthogonal but as good as they can be within the limitation of the hardware and software. The reduction in iteration count for low thresholds shows that an ILU factorisation is very good at meeting the requirements of approximating $\mathbf{A}^{-1}$. However, a low threshold also means that the factorisation has a low sparsity level and as such specialist sparse techniques cannot be exploited.

Additionally, the low threshold level means that the computational cost of calcu-

Threshold	Model 1 $N = 136$		Model 2 $N = 184$	
	Iteration Count	Sparsity (%)	Iteration Count	Sparsity (%)
1×10^{-4}	2	99.43	3	98.29
1×10^{-3}	4	97.15	4	93.50
1×10^{-2}	9	66.99	10	63.23
1×10^{-1}	FAIL	7.93	FAIL	8.39
1	FAIL	2.14	FAIL	3.42
10	FAIL	0.90	FAIL	1.31
20	FAIL	0.87	FAIL	1.04
30	FAIL	0.80	FAIL	1.00
35	38	0.77	FAIL	0.96
40	35	0.74	40	0.57

Table 7.1: Iteration count and sparsity levels for ILUT preconditioned GMRES

lating the factorisation is high. The effect of non-exact arithmetic and its resultant effect on computational cost is shown in figure 7.4. This shows the variation in

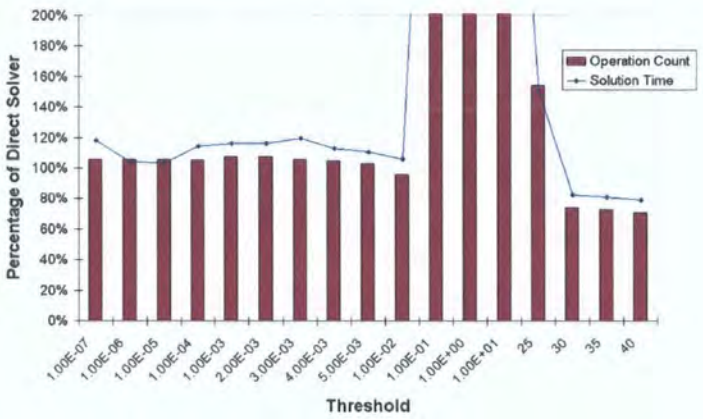


Figure 7.4: Computational cost of ILUT preconditioning strategy

computational cost and actual solve time with respect to threshold level normalised with respect to a standard direct solver of the same problem. From this it can be seen that the computational cost of calculating the ILU factorisation causes the overall computational cost to be more than a direct solver until the threshold is high

enough that the ILU is approximately a diagonal preconditioner.

Thus, the ILU preconditioner meets two of the three criteria for a good preconditioner in that it approximates $\mathbf{A}^{-1}$ well at low thresholds and is computationally reasonable to apply but at low thresholds it is not computationally cheap to calculate. At high thresholds the technique approximates a diagonal preconditioner but is computationally more expensive. As a result an alternative is required which is effective at reducing the iteration count in a similar manner to the ILU factorisation but does not require the computational cost of the ILU factorisation.

To help guide the selection of a suitable preconditioner it is important to note the facilities available to the preconditioner that can be exploited in the application of the preconditioner. There are four main points to note.

1. The problems considered are small
2. The computer has a large quantity of RAM available
3. The focus of the work is rapid reanalysis
4. Modern operating systems support multi-threading of processes

By exploiting these factors it is possible to produce a new scheme for preconditioning small problems under reanalysis situations.

7.4 Proposed scheme

As the problems considered in this work are relatively small problems then storage of a factorisation is not a critical parameter. As a result it is not a requirement that the preconditioner is sparse. Additionally it has been found that an ILU threshold based factorisation is effective at reducing iteration count as long as a low threshold is employed. Thus if a preconditioner can exploit the effectiveness of the ILU threshold based factorisation at reducing iteration count whilst reducing the computational overhead associated with calculating the factorisation then it will be a computationally efficient preconditioner.

As a result a new scheme, outlined in figure 7.5, is proposed for the initial analysis and subsequent reanalysis of problems. By exploiting the fact that a direct solver

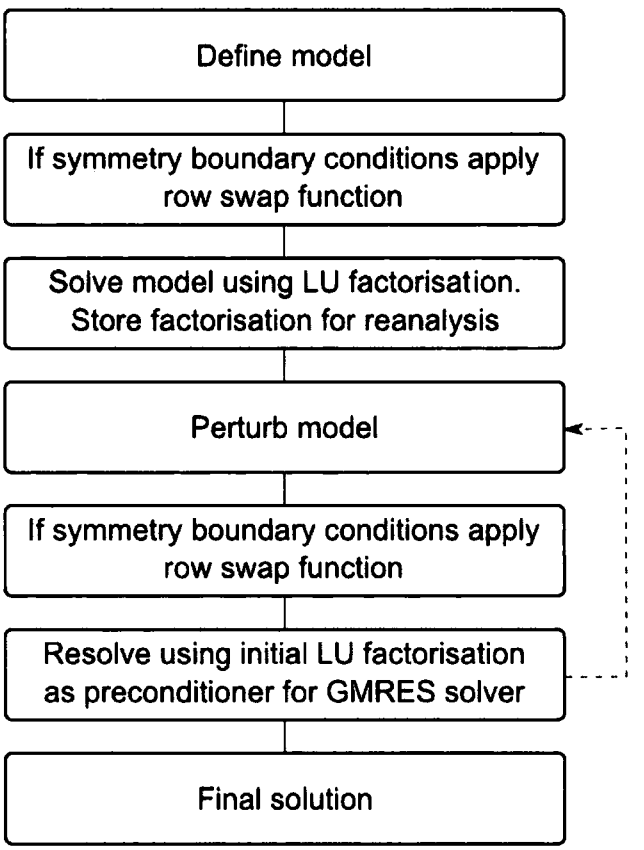


Figure 7.5: Proposed analysis and reanalysis scheme

is used for the initial solution we can extract and store information for use in later reanalysis. In particular it is possible to employ an LU preconditioner as the initial solver and store the **L** and **U** factors. These factors can be then implemented as a preconditioner to the perturbed system in the reanalysis phase. As the factors are already calculated they are available at zero cost to the reanalysis routine and so the only computational cost associated with the preconditioner is its application in each iteration. As the **L** and **U** factors closely approximate those for the perturbed system they will provide a very efficient preconditioner, causing the eigenvalues to cluster and, as a result, reduce the overall iteration count.

A drawback of this scheme is that the preconditioner is based on the originally solved matrix system and, as a result, has the potential to deteriorate in efficiency as multiple perturbations occur; this is similar to the drawback faced by Kane *et al.* (1990) and Leu (1999). This deterioration in the preconditioner will have the effect of increasing iteration count and the computational cost of the iterative scheme. This can be seen by the spread of the eigenvalues after multiple perturbations. Figure 7.6 is the initial state of a simple BEM problem with eigenvalues given by figure 7.7(a), additionally the geometry after 1, 5 and 10 perturbations is identified within the model. Figure 7.7(b) displays the effectiveness of the LU factorisation at clustering the eigenvalues together around unity. Figure 7.7 shows the deterioration of the

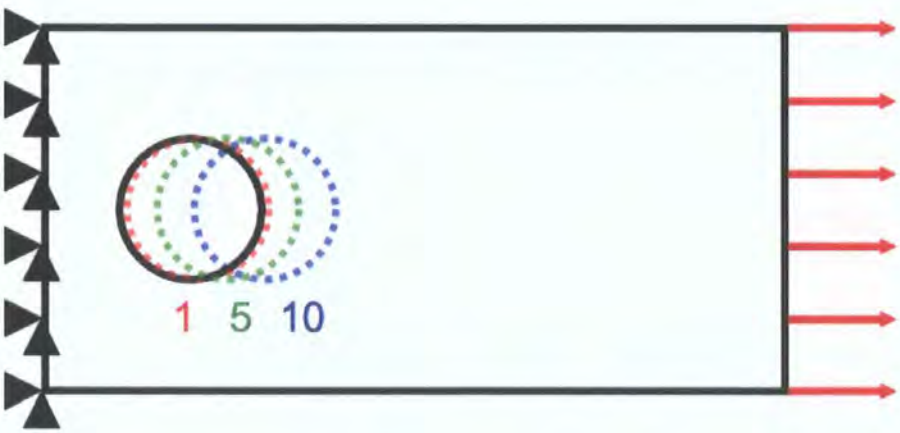
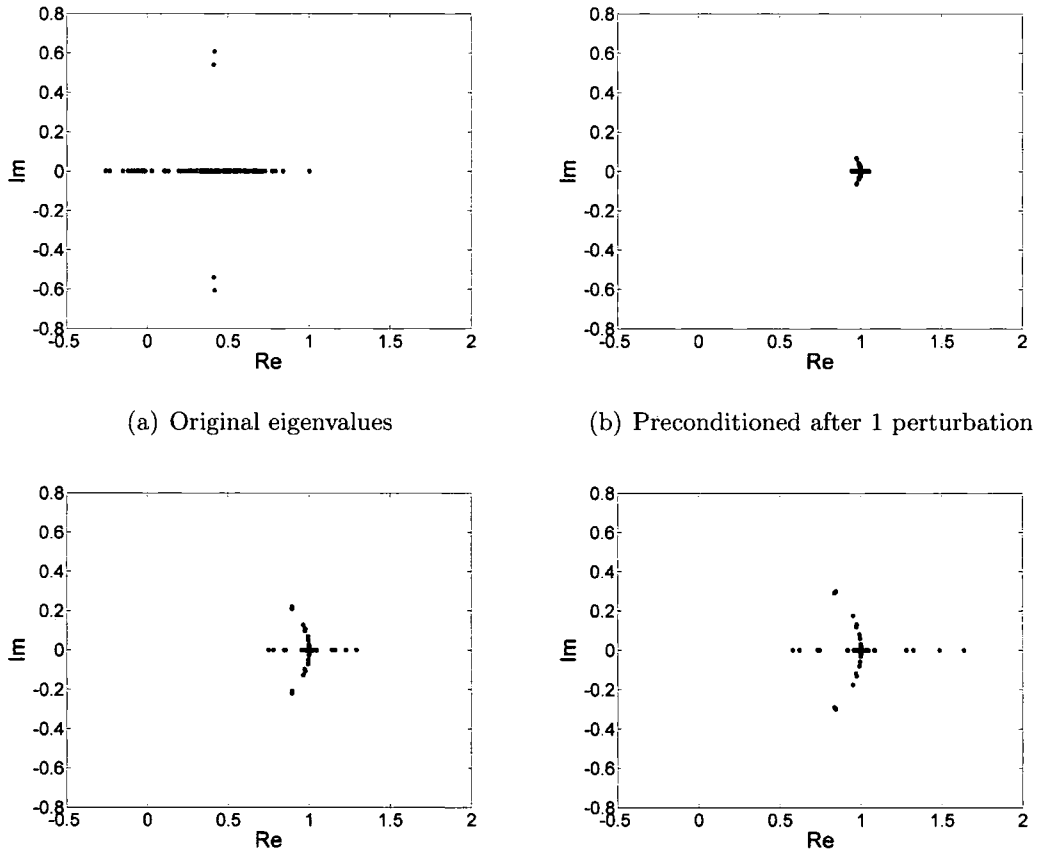


Figure 7.6: Initial model

LU factorisation as a preconditioner over multiple perturbations.

As a result it is desirable to employ an update strategy on the preconditioner such



(a) Original eigenvalues

(b) Preconditioned after 1 perturbation

(c) Preconditioned after 5 perturbations

(d) Preconditioned after 10 perturbations

Figure 7.7: Deterioration of the LU factorisation after multiple perturbations

that a recent form of the **L** and **U** factors are available for the reanalysis function. Updating of the preconditioner will maintain the clustering effect identified in figure 7.7(b). To allow the updating of the preconditioner effectively and efficiently a separate thread will be initiated. To ensure that the thread operation does not interfere with the higher priority task of user interaction, it is necessary to enforce a priority level when initialising the thread. Thread priority can be defined from highest to lowest priority as one of the following values:

- Realtime
- High
- Above normal
- Normal
- Below normal
- Idle

By default, threads are initialised with a normal priority. As the thread to update the preconditioner is less important than the main program thread, so that it does not interfere with the normal operation of the program, it will be initialised with a thread priority of *idle*. Thus it will only become active during periods when the central processing unit (CPU) has no other tasks scheduled. In multi-core machines this issue is negligible as the main program thread and preconditioner thread can run in parallel on separate cores.

Due to the deterioration of the effectiveness of the preconditioner and the requirement of the update scheme, it is necessary to determine the rate of deterioration such that an appropriate update rate is achieved. By profiling* the specific routines used within both the factorisation and the iterative solver it is possible to determine how often it is necessary to update the preconditioner.

*By profiling functions the performance will be degraded and as a result it is not possible to determine absolute timings for functions. However, as the degradation will be consistent it is possible to compare functions within individual profile sessions.

There are two main routines which need to be considered,

- The preconditioned GMRES solver
- Updating of the **L** and **U** factors

These functions can be further divided,

- GMRES - Set-up
- GMRES - Iteration
- Update LU - Locked stage
- Update LU - Factorise

Set-up costs associated with the GMRES solver cover the initialisation of various variables, the necessary destruction of variables and the extraction of the solution from the GMRES iterates, thus these costs are only incurred once per solution. Costs associated with the iteration phase of the solver are recurring costs associated with the loop between lines 2 and 10 in algorithm 5.9. Updating of the preconditioner can be split into two sections.

1. Locked sections.
2. Unlocked sections.

Locked sections are surrounded by software mutexes (mutual exclusive locks) which prevent other threads of execution accessing the variables associated with the mutexes. This is important as it prevents two separate threads attempting to access the same variable and altering the contents. In the secondary update thread the locked section is used to copy the **A** matrix into a local copy, the **A** matrix is then unlocked so that the main program thread can still access it while the secondary thread can factorise the local copy, finally the preconditioning matrix is locked and the new **L** and **U** factors are copied back to the main dataset for use in reanalysis. Thus, the locked stage of updating the preconditioner allows the thread-safe copying of variables between threads, whilst the factorisation runs in an unlocked state as there is no inter-thread communication.

Table 7.2 lists the appropriate timings for the two main routines; the GMRES iterative solver and updating the LU preconditioner, for a sample problem with $N = 184$.

Function Name	Profile time (μs)
GMRES - Set-up	13131
GMRES - Iteration	4862
Update LU - Complete	171756
Update LU - Locked Stage	6054
Update LU - Factorise	165702

Table 7.2: Function profiles for updating of LU factorisation

Thus on multi-core systems, in which the main program thread and the preconditioner update thread are able to run concurrently, the cost for updating the preconditioner is the period for which the main program thread is locked from altering the **A** matrix, 6054μs in the illustrative example included here. Thus, if the number of iterations to solve a perturbed model increases by two iterations then it will be efficient to update the preconditioner. The test case selected for this example was simply moving a hole in a plate, and so the boundary solution in reanalysis is likely to be approximated well by the previous solution. The model was also a very small one. Other, larger, example models are almost certainly going to be more sensitive to the deterioration of the preconditioner and require a greater number of iterations. Hence, greater emphasis must be placed on updating the preconditioner. It is therefore concluded that the **L** and **U** factors should be updated on every perturbation.

For single-core machines the problem becomes more complex. If the reanalysis is performed in a non real-time format then the cost of updating the preconditioner is simply the period for which the **A** matrix is locked, 6054μs. However, if the reanalysis is run in a real-time format then the LU factorisation thread will not be allocated any time on the CPU due to having an idle priority level. It should be noted that if the user pauses, and as a result reduces the CPU load, then the secondary thread will be given time on the CPU and allowed to proceed. Moreover,

as the *profiled factorisation* takes 0.165s[†] then a pause of this duration will allow the processor to update the preconditioner.

Based on these scenarios it is sensible to update the preconditioner as often as the computer will allow it such that the most recent version of the factorisation is available. This is due to the relatively small perturbation required to increase the iteration count and, as shall be seen in chapter 8, the outstanding effectiveness of a recently updated preconditioner at reducing iteration count.

7.5 Concluding remarks

In this chapter a new scheme for the analysis and, in particular, reanalysis of small two-dimensional problems has been proposed. The scheme incorporates the use of an LU factorisation as the direct solver for the initial problem; these factors are then stored and recalled during the reanalysis process as a preconditioner for the perturbed system.

As the preconditioner is based on the original factorisation an update scheme has been introduced. The update scheme employs a second, low priority, thread such that updating the preconditioner will not interfere with the primary role of the software and will only run when there is idle time on a CPU core. The update scheme has been analysed and it is proposed to update the preconditioner as often as the CPU will allow. Thus multi-core machines will benefit in particular as the secondary thread can run on a separate core to the main program thread.

[†]As this is the profiled case this is an exaggerated time and is expected to be below 0.1s for an unprofiled analysis.

CHAPTER 8

Results

Chapters 6 and 7 have introduced techniques for accelerating the solution of boundary element models of elastostatic problems. In this chapter results will be presented demonstrating the effectiveness of the proposed solution for the problems of interest. Additionally, limiting cases will be presented to give an upper bound on the problem size that allows real-time solution of the problem on current hardware.

To allow a comparison between the proposed methodology and alternative techniques it is necessary to *profile* the necessary functions under analysis. Profiling is a technique that allows the comparison of timings within separate functions or blocks of code. This allows functions that give rise to bottle necks within the programs process to be targeted and optimised. Alternatively, profiling can be used to allow the comparison of techniques to conclude which technique is more efficient for a particular task. The introduction of profiling incurs additional costs and, as a result, the figures produced can only be used for comparative purposes. Thus, the integration schemes proposed in chapter 6 have been compared with an adaptive Gauss-Legendre quadrature scheme, and the equation solution proposed in chapter 7 has been compared with a direct solver and a diagonally preconditioned GMRES iterative solver.

A personal computer with a Pentium M 2GHz processor and 2GB of RAM was used in the profiling study, using Microsoft Visual C++ .NET.

8.1 Implementation

The process of re-analysis on a design change has been accelerated using a variety of techniques. However, the aim of this work is to enable stress contours to update dynamically as the model's geometry is modified by some interactive operation.

The Windows operating system sends messages to instances of a program running in a window. In particular, the message we use here is the WM_MOUSEMOVE message that is sent when the mouse is moving in the active window. Upon receipt of this message, if contours are being displayed and a geometric change is being performed, the program automatically remeshes the changed geometry, ideally with the same number of elements, and initiates a re-analysis, finally updating the contour display.

It has been found that the speed of the re-analysis is sufficient to enable this type of dynamic display. However, for best performance it is recommended to initiate a re-analysis only once every two times the WM_MOUSEMOVE message is received. This provides a suitably fast update and achieves the aim of smoothly updating stress contours.

It is finally worth mentioning one more point about implementation of LUT integration approaches with respect to different levels of memory cache. The memory footprint of the LUTs is very much greater than the size of the rapid access memory cache. As a result, each time an LUT value is extracted it is likely that the run-time cost is incurred for a retrieval from a random memory address. However, the performance can be enhanced by a factor of four by structuring the LUTs in RAM so that values for the different matrix terms, for the same R_m and $(\phi - \theta)$, appear in adjacent memory locations. Processors normally retrieve information from remote memory locations by bringing 64 byte blocks into the rapid access cache; a space sufficient for four floating point numbers. In this way, when the first LUT value is required we incur the cost of retrieval from a random memory address, but when

we subsequently require the next three LUT values for the different matrix terms they are available in the rapid access cache for an insignificant cost. Without this structure to the LUT, the method would be less efficient even than a Gauss-Legendre scheme.

A similar ordering strategy is used within the surface fit implementation for storage of basis functions.

8.2 Integration

It has been seen in chapter 4 that complicated integrals that can not be solved analytically may be integrated numerically. In the current work, the storage of precomputed integrals has been used to accelerate the computation by reducing the necessary computational cost. Two techniques have been proposed for the storage of precomputed boundary element integrals.

1. Look-up tables.

2. Surface fit equations.

These results will be presented separately and then compared against a Gauss-Legendre scheme. It should be noted that the Gauss-Legendre scheme is highly optimised, using an efficient RISP algorithm (Kane *et al.*, 1989) and using only a 2^{nd} order Gauss-Legendre scheme with reuse of Jacobians for the majority of the integrations. Timings can be scaled to provide an approximate account for implementations that use a predominantly higher order of Gauss-Legendre quadrature.

Figure 8.1 shows the example problem used in the analysis of the integration phase. A rectangular plate with a circular hole containing 46 quadratic elements is used, so that it contains a mixture of flat elements and circular arc elements. However, no further details of the problem are presented here nor is the mesh shown in the figure, because the results are presented in terms of the mean time taken to perform each integral, so the precise geometry and model size become irrelevant. It is sufficient to note the mix of 34 flat and 12 circular arc elements. The internal point

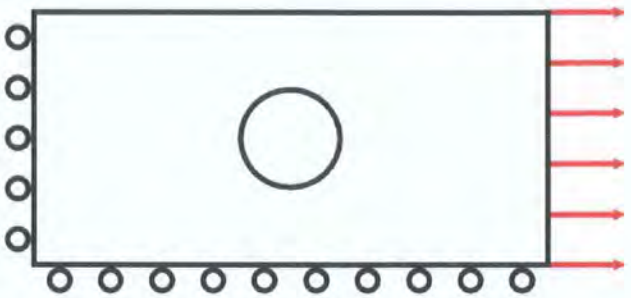


Figure 8.1: Analysis model for integration comparison

solution was isolated for this profiling study, but the timings are equally applicable to the integration in the matrix assembly phase of the BEM solution.

8.2.1 Look-up tables

The use of LUTs falls into two main categories.

- Non-Interpolated LUTs.
- Interpolated LUTs.

The type of LUT chosen for each element type, flat or circular arc elements, will affect the overall performance of the technique but will also alter the memory requirement for storing the LUT.

The results presented in this section have a degree of variability on account of the nature of profiling software. This variability will be investigated in section 8.2.2.

Table 8.1 shows the relevant timings for the use of non-interpolated LUTs to calculate all 60 terms for the displacement and stress boundary integral equations as required for quadratic elements. As they are not interpolated these are faster but at a much higher memory cost, 2.89GB, and as a result are not feasible with current technology. Timings in table 8.1 are stated on a per integration basis. Additionally, timings for an adaptive Gauss-Legendre scheme are presented for comparison.

The use of interpolated LUTs increases the computational cost as it is necessary to extract two values from the LUT and interpolate between them for every term required. This is a result of the interpolation only being required in the R_m parameter. Table 8.2 shows the additional cost required for the interpolation. However,

Technique	Mean run time (μs)
Adaptive Gauss-Legendre	57.8
LUT - Flat element	30.6
LUT - Circular arc element	31.5

Table 8.1: Timings for integration using non-interpolated LUTs

interpolation leads to a much reduced memory requirement of 102MB.

Technique	Mean run time (μs)
Adaptive Gauss-Legendre	57.8
LUT - Flat element	34.5
LUT - Circular arc element	35.6

Table 8.2: Timings for integration using interpolated LUTs

As a result it would seem prudent, based on timings in tables 8.1 and 8.2, to employ only non-interpolated LUTs. However, as noted in chapter 6 the memory requirement makes this impractical with current hardware.

To optimise the LUT technique for small everyday problems a scheme that utilises both non-interpolated LUTs for flat elements and interpolated LUTs for circular arc elements is proposed. Table 8.3 shows the relevant timings for the use of LUTs under the proposed scheme.

Technique	Mean run time (μs)
Adaptive Gauss-Legendre	57.8
LUT - Flat element	30.6
LUT - Circular arc element	35.6

Table 8.3: Timings for integration using LUTs

From these timing it can be seen that the use of LUTs outperform adaptive Gauss-Legendre quadrature by 38% for interpolated LUTs and by 47% for non-interpolated LUTs.

The difference in timing between flat and circular arc elements shown in tables 8.1 and 8.2 is due to the additional computational cost of ensuring that a non-flat element is suitable for use with the circular arc LUTs.

8.2.2 Variability of profiling

Profiling of software contains a number of variables which can influence the resultant timings. These include factors such as,

- Number of processes running.
- Type of processes running. Typically, *system* processes will run at a higher priority than *user* processes and as a result a system process may interrupt a user process, causing a degradation in performance.

Thus, it is necessary to assess the variability within the profiling operation.

Profiling of the Concept Analyst software has been performed 20 times for both interpolated and non-interpolated LUTs; the data for the profile runs are listed in appendices M and N. From these data it is possible to extract appropriate means and standard deviations, listed in tables 8.4 and 8.5. Figures 8.2 and 8.3 display

Technique	Mean run time (μs)	Standard Deviation
Adaptive Gauss-Legendre	57.8	0.78
LUT - Flat element	30.6	0.48
LUT - Circular arc element	31.5	0.55

Table 8.4: Variability parameters for non-interpolated LUTs

Technique	Mean run time (μs)	Standard Deviation
Adaptive Gauss-Legendre	57.8	0.78
LUT - Flat element	34.5	0.33
LUT - Circular arc element	35.6	0.53

Table 8.5: Variability parameters for interpolated LUTs

box and whisker plots representing the spread of data from the analyses. The lower and upper box limits are set to the 10th and 90th percentile respectively. From these figures it can be seen that the variability within the profiling data is low and as a result the factors mentioned previously have a limited effect on the variability.

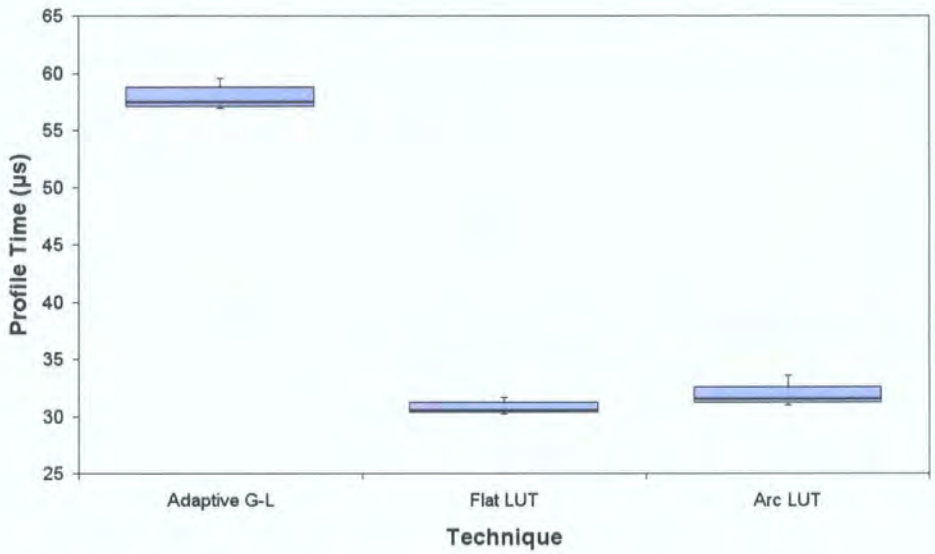


Figure 8.2: Spread of profiling data for non-interpolated LUTs

8.2.3 Surface fit equations

Surface fits have only been implemented for flat elements. In order to accelerate the computation of the surface fits, values for each basis function ($\sin(q\theta)$, $\cos(q\theta)$ etc) have been precomputed at initial start-up and stored such that they can be easily extracted at computation time. Thus the memory requirement for this technique is 5.8MB.

Technique	Mean run time (μs)
Adaptive Gauss-Legendre	57.9
Surface fit ($\phi = 0°, 90°, 180°, 270°$)	15.1
Surface fit (arbitrary ϕ)	27.4

Table 8.6: Timings for integration using surface fit equation

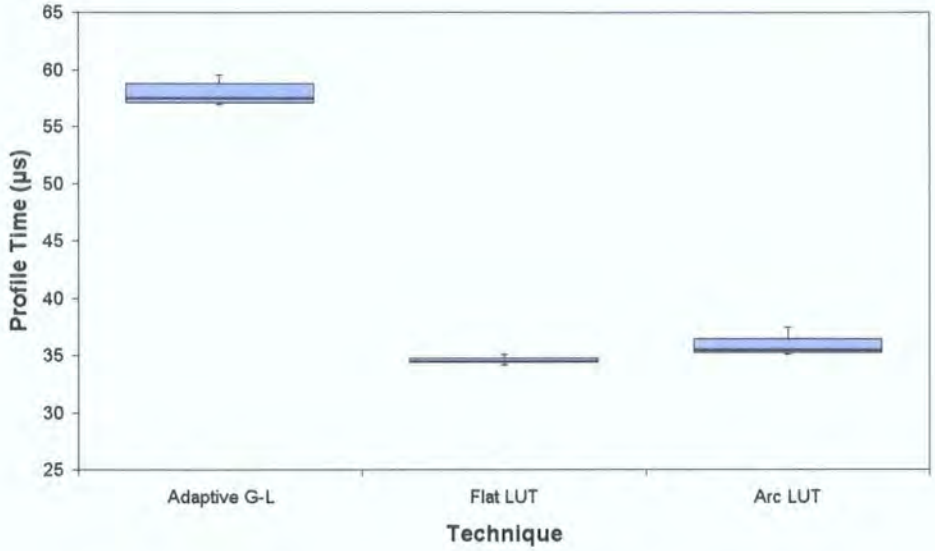


Figure 8.3: Spread of profiling data for interpolated LUTs

Table 8.6 displays timings for the integration using surface fits for two cases. The first case is for particular values of ϕ, indicating integration over horizontal and vertical elements. This has been implemented as the problems of particular interest are, almost without exception, formed from an initially rectangular shape and, as a result, a large proportion of the elements will be applicable to the surface fits implemented. It can be seen that by hard-coding these particular surface fits of interest the computational cost is reduced by 74% over conventional Gauss-Legendre quadrature. However, this reduces the applicability of the method to a proportion of the elements within the model.

To overcome the limitation imposed by calculating surface fits for specific values of ϕ it is possible to implement a coordinate transformation scheme as used within the LUT scheme. In this scheme it is only necessary to calculate surface fits for $\phi = 90°$ and then apply the coordinate transformation given in equation (6.19) for the displacement boundary integral equation. The additional cost of implementing the coordinate transformation has been calculated as 12.3μs. Hence, the cost of implementing surface fits for arbitrary ϕ becomes 27.4μs. This outperforms the variable Gauss-Legendre quadrature scheme by 53%. Table 8.7 details the computational cost for the example given in figure 8.4. From this it can be seen that

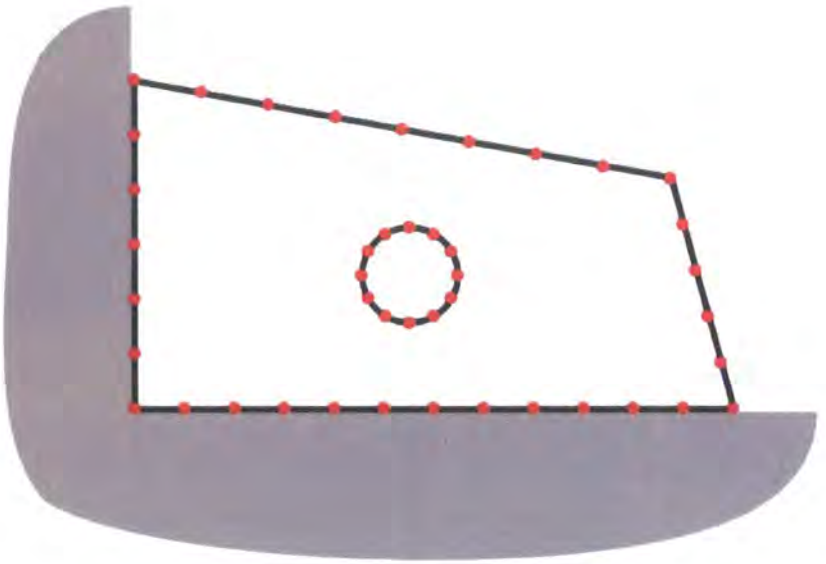


Figure 8.4: Example of arbitrary element orientation saving

although the computational cost is higher on a per integration basis when coordinate transformations are implemented, as more elements can be integrated using the proposed technique then the overall computational cost is lower.

Element Type	Fixed ϕ	Arbitrary ϕ
Applicable element	4.68	14.57
Non-applicable element	25.00	12.00
Total Cost	29.68	26.57

Table 8.7: Comparison of timings for fixed ϕ and arbitrary ϕ surface fits

8.2.4 Comparison of integration techniques

Two main factors need to be considered when comparing the proposed integration techniques with current employed techniques.

1. Computational cost.

2. Memory cost.

The main aim of this work is the minimisation of the computational cost subject to current constraints provided by memory. As a result it is necessary for both of these points to be considered together. This is a result of the ability of operating systems to use not only RAM (very fast memory) to store items but also to use swap space (specially defined space on a hard disk drive, potentially bigger than RAM but significantly slower). Thus, if a process requests more memory than is currently available in RAM the operating system will designate some of the RAM to the process and then a proportion of swap space. Swap space can not be accessed directly by the CPU but the values must be loaded in and out of RAM (hence the term swap space) when accessed. This process causes a significant delay to the access of values. As the aim of this work is to deal with real-time analysis and updating of contours the accessing of swap-space is unfeasible.

Technique	Mean run time (μs)	Memory Cost (GB)
Adaptive Gauss-Legendre	57.8	0.000
Int. LUT - Flat element	34.5	0.029
Non-int. LUT - Flat element	30.6	0.818
Int. LUT - Circular arc element	35.6	0.071
Non-int. LUT - Circular arc element	31.5	2.072
Surface fit ($\phi = 0°$, 90°, 180°, 270°)	15.1	0.006
Surface fit (arbitrary ϕ)	27.4	0.006

Table 8.8: Summary of integration timings and memory requirements

Table 8.8 displays a summary of run time and memory requirement for each of the proposed techniques. All timings have been made assuming that there is enough RAM to store the LUTs, running programs and operating system entirely without the need for swap space. From table 8.8 it can be seen that the use of non-interpolated LUTs for circular arc elements is currently unreasonable due to the high memory requirement. These LUTs alone would require the use of swap space for storage causing the run time to be increased dramatically and preventing real-time analysis. However, it is expected that the memory available on PCs will

increase, and as a result it will be necessary to re-evaluate this stance as technology improves.

The remaining techniques do not have the same limitation due to the much reduced memory requirement. Thus, computational cost becomes the deciding factor. Based on this, if a model is dominated by flat elements orientated with the horizontal and vertical then the use of specific surface fits produces the best performance gain. Additionally, if particular orientations of element are popular, for example $\phi = 45°$, then surface fits can be generated for these particular values of ϕ extending the benefits to these elements.

Figures 8.5 and 8.6 show the relative performance of the surface fit technique as the number of applicable elements varies. As expected, the relationship between the number of applicable elements and the overall time saving is linear. Figures

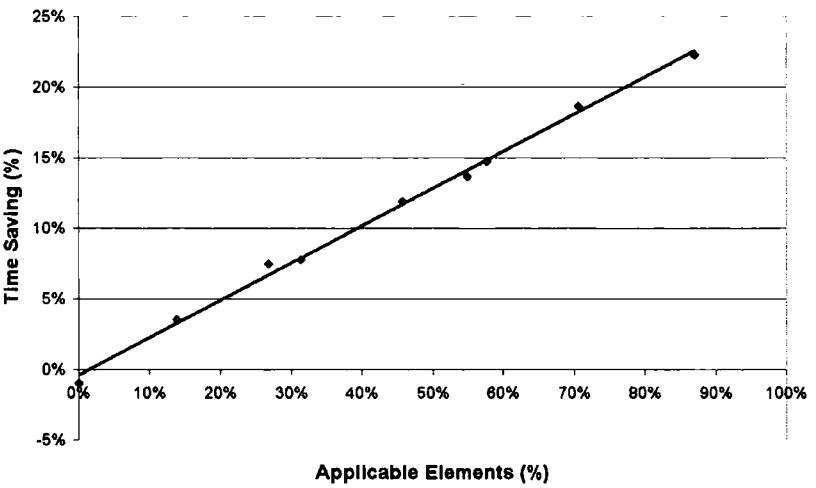


Figure 8.5: Time saving as number of applicable elements is varied - Boundary terms

8.5 and 8.6 include offsets from the origin for two reasons. Firstly, the use of fast integration techniques such as those presented within this work rely on particular types of element. As a result there is additional computation required to ensure that an element is of a suitable type, and this will lead to a slight computational overhead to the technique. The second reason is a result of variability within profiling of the functions. This will have the effect of spreading the data around the relationship.

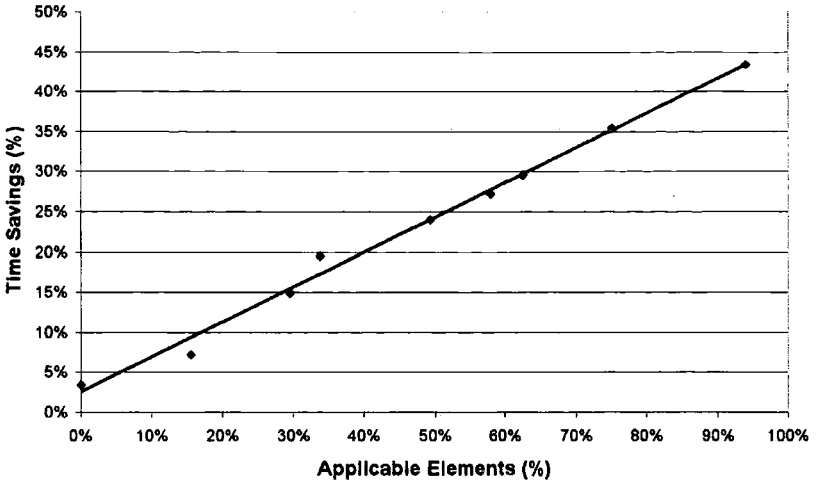


Figure 8.6: Time saving as number of applicable elements is varied - Internal points

However, if elements are arbitrary in orientation then the use of a coordinate transformation, at additional computational cost, is necessary. Use of surface fits with a coordinate transformation is computationally cheaper than the associated LUT method but is only currently applicable to flat elements. Extension of surface fits to circular arc elements should present few technical challenges, but remains a subject for further investigation.

8.3 Equation solution

It has been seen in chapter 5 that systems of linear equations may be solved using a variety of techniques. In the current work, the emphasis on the solution within the reanalysis problem has focused attention on iterative methods, specifically a pre-conditioned GMRES algorithm, as the approach of choice. The reader is reminded at this stage that we benefit from a good first approximation to the solution vector and, furthermore, the algorithm proposed in chapter 7 makes available an LU decomposition of an approximation to the matrix **A**.

Computational efficiency of the solution of the matrix equation,

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{8.1}$$

can be defined by two main parameters.

1. Iteration count.

2. Overall computational time.

Iteration counts should not be compared between different form of preconditioning without additional information due to the lack of detail regarding the computational cost of calculating and applying the preconditioner. The greater the degree of sparsity of the preconditioning matrix, the greater the computational efficiency in its application at each iteration. However, comparison of runs with the same preconditioner, for example investigating different degrees of perturbation or different types of perturbation, is a fair comparison as the preconditioner will have the same computational cost to calculate and apply at each iteration.

Figure 8.7 shows five models used to test the proposed equation solution technique for a variety of perturbations. Three main classes of perturbation have been considered.

1. Moving a point on an object.

2. Moving a shape (for instance a circle that represents a hole in the model).

3. Resizing of a fillet.

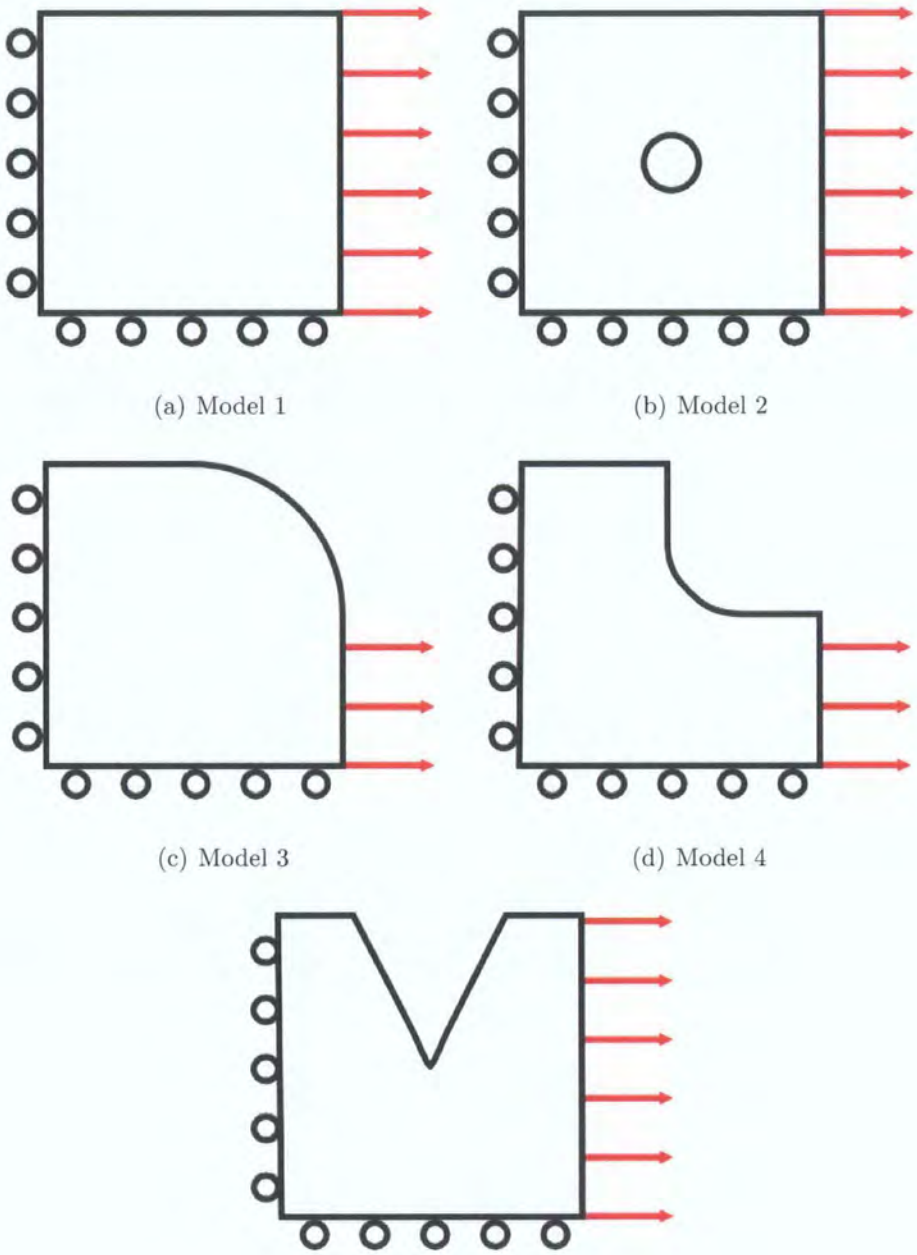
These three perturbations are the main perturbations implemented by users that can be solved using reanalysis techniques. The resizing of a fillet has been further divided into three classes exhibiting different severity of stress concentration.

1. Convex fillet.

2. Concave right angle fillet.

3. Concave acute angled fillet.

The fillets have been resized both increasing the radius of the fillets and decreasing the radius of the fillets. Consideration of both increasing and decreasing is important as the initial stress pattern affects the initial solution within the iterative solver and as a result can affect the rate of convergence. Moreover, as the fillet radius is varied elements can be moved or transferred from the fillet to adjacent lines on



(a) Model 1

(b) Model 2

(c) Model 3

(d) Model 4

(e) Model 5

Figure 8.7: Models used to analysis proposed equation solution technique

the boundary, or vice versa. This allows the total number of elements within the model to remain constant whilst ensuring that the elements are suitably placed. The transferring of elements between boundary lines causes a distinct change within the matrix equations.

In chapter 7 the use of an approximate but complete LU preconditioner has been proposed for problems of reanalysis. The models presented in figure 8.7 will be considered and the iteration counts for resolution using no preconditioning, diagonal preconditioning and a complete but approximate LU preconditioner presented in table 8.9. The low iteration counts for the complete but approximate LU precondi-

Preconditioner	Perturbation Type			
	Move Point	Move Shape	Ext Fillet Resize	Int Fillet Resize
None	30 - 50	31 - 49	36 - 48	34 - 53
Diagonal	39 - 53	36 - 47	36 - 44	43 - 50
Full LU	2 - 17	2 - 9	3 - 9	3 - 13

Table 8.9: Summary of iteration counts for various preconditioners

tioner are a result of the preconditioner being close to the inverse of the perturbed matrix, $\mathbf{A}'$. Moreover, as there is a zero computational cost to calculating the preconditioner, and the application of the preconditioner is $\mathcal{O}(N^2)$ but is only applied a small number of times, the overall computational cost and hence run-time is low. Normalising the computational cost with respect to a direct solver is represented by figure 8.8. This shows the effectiveness of the proposed equation solution technique for cases of non real-time analysis i.e. the problem has been solved, a perturbation is made to the model and then the reanalysis occurs. This shows that the use of a complete but approximate LU factorisation is effective at reducing the overall computational cost of solving the matrix system even for both relatively large geometric perturbations and perturbations in which a large number of elements are altered. By way of putting these results into context, a diagonal preconditioner typically reduces the normalised solution time to 70% of a direct solver for small everyday problems.

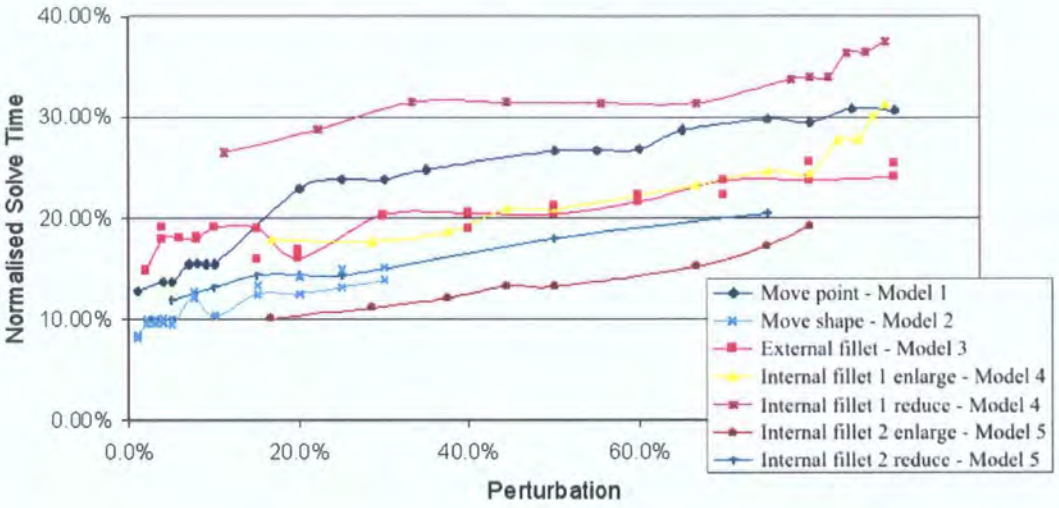


Figure 8.8: Normalised solve time for a variety of perturbations (Trevelyan *et al.*, 2004)

The aim of this work is to perform real-time reanalysis of elastostatic problems and as a result the geometric perturbations involved will typically be less than 20%. On account of this small geometric perturbation it can be surmised that the variation in displacements and tractions will similarly be relatively small leading to rapid convergence from the first approximation.

Typically, only a relatively small part of the model will be perturbed at each user interaction. Thus most perturbations will involve less than 30% of the total number of elements in the model, but might potentially be as high as 50%. Thus the typical saving within the equation solution can be seen to vary between 75% and 85%.

8.4 Overall strategy

The techniques proposed within this thesis provide the ability of real-time elastostatic analysis of small two-dimensional problems. Moreover, the techniques can be applied to larger problems as an acceleration technique although it is clear that, for any given computational resources, there will be an upper bound on problem

size beyond which the re-analysis will fail to have a real-time character. As a result, although the initial target market is small two-dimensional problems, the techniques proposed have advantages for larger problems in both two and three dimensions.

8.4.1 Problem size

As the problem size increases, the overall computational cost associated with both integration and the solution of the matrix equations increases. As a result any method aimed at producing a real-time solution will have a maximum problem size for which real-time analysis is possible.

The computational cost of solving the matrix equation given by equation (8.1) is related to the overall problem size and, to a lesser extent, the number of computer cores on the personal computer (a multi-core will be able to update the preconditioner more often than a single core machine, thus improving performance). A number of variables affect the computational cost of the integration routine.

- Overall problem size.
- Number of elements to which the integration scheme can be applied.
- Number of elements being perturbed in each step.

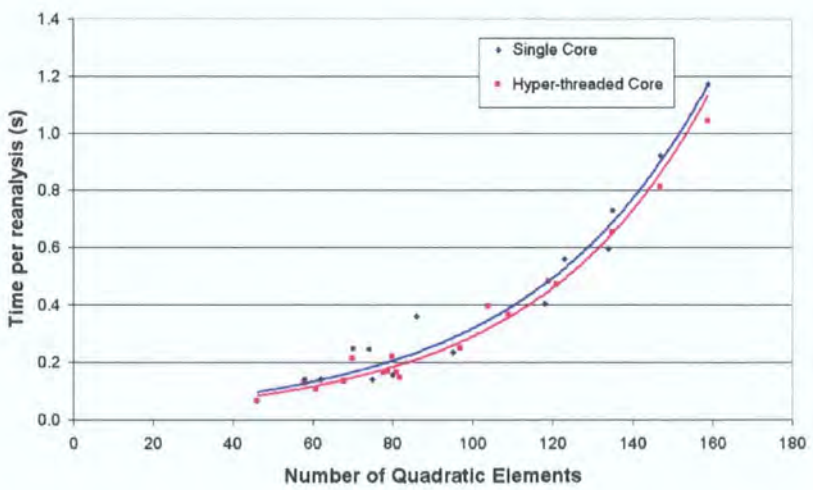


Figure 8.9: Variation in reanalysis time with problem size

Figure 8.9 shows the variation in reanalysis time for a variety of problem sizes under a single core machine and a hyper-threaded core machine*. As the problem size increases the computational cost per reanalysis increases. This is a result of the additional integrations required to prepare the matrix equations for solution, and additionally a larger size problem will typically take longer to solve with an iterative solver.

Additionally figure 8.9 shows the performance for a hyper-threaded enabled core (Pentium 4 3.6GHz processor). It is difficult to compare these times directly as the CPU cores run at different clock speeds and are of different general architectures. It is the author's belief that the use of a hyper-threaded core will enable a small performance gain over a single core machine but will also allow a user to have additional applications open whilst performing an analysis. If a true multi-core machine were utilised it is the author's belief that this performance gain would be larger as a result of the preconditioner being updated on the separate core.

It is the author's experience that a problem consisting of at maximum 85 quadratic elements can be analysed in a real-time manner such that contour plots are updated as the geometry is perturbed. Lagging within the update of contours occurs when larger problems are analysed.

8.5 Concluding remarks

In this chapter results have been presented showing the effectiveness of the proposed techniques. Additionally, results have been presented showing limitations with respect to problem size for both single and dual core machines. This shows the advantage that multi-core machines have for this implementation.

Details on implementation, to optimise computational cost and usability, have also been presented within this chapter for both the integration and equation solution schemes.

*Hyper-threading is an Intel specific technique for utilising idle components on a single CPU core in separate threads. As only a proportion of the components are replicated it can pretend to be a separate core but is not in the true multi-processor definition.

CHAPTER 9

Extension to Other Application Areas

The techniques proposed within this thesis have currently only been applied to elastostatic problems being analysed with the boundary element method. In this chapter a number of extensions will be considered for applying the proposed techniques to different forms of analysis.

Extensions will be considered with respect to the two areas targeted for acceleration within this thesis,

1. Integration.

2. Equation Solution

9.1 Integration

Techniques for accelerating the integration phase can be extended to different applications of the boundary element method by consideration of the appropriate fundamental solutions. Additionally, consideration of three dimensional problems is presented.

9.1.1 Potential flow

Heat transfer for boundary element formulations has been presented in chapter 3. The boundary integral equation for potential flow is given by (Becker, 1992),

$$\phi(\kappa) + \int_{\Gamma} K_1(\kappa, Q)\,\phi(Q)\,\mathrm{d}\Gamma(Q) = \int_{\Gamma} K_2(\kappa, Q)\,\frac{\partial\phi(Q)}{\partial n}\mathrm{d}\Gamma(Q) \tag{9.1}$$

where ϕ is the potential and K_1 and K_2 are the fundamental solutions and are given by

$$K_1(\kappa, Q) = \frac{1}{2\pi r(\kappa, Q)} \tag{9.2}$$

$$K_2(\kappa, Q) = \frac{1}{2\pi}\ln\left[\frac{1}{r(\kappa, Q)}\right] \tag{9.3}$$

Thus, discretising equation (9.1),

$$\phi(\kappa) + \sum_{elem}\int_{-1}^{+1} K_1\mathbf{N}^{\mathbf{T}}J(\xi)\,\mathrm{d}\xi\boldsymbol{\phi} = \sum_{elem}\int_{-1}^{+1} K_2\mathbf{N}^{\mathbf{T}}J(\xi)\,\mathrm{d}\xi\frac{\partial\boldsymbol{\phi}}{\partial\mathbf{n}} \tag{9.4}$$

LUTs can now be created of the form

$$h^{LUT} = \frac{1}{2\pi}\int_{-1}^{+1}\frac{1}{R_m}\mathbf{N}^{\mathbf{T}}\frac{J(\xi)}{L}\mathrm{d}\xi \tag{9.5}$$

$$g^{LUT} = \frac{1}{2\pi}\int_{-1}^{+1}\ln\left[\frac{1}{R_m}\right]\mathbf{N}^{\mathbf{T}}\frac{J(\xi)}{L}\mathrm{d}\xi \tag{9.6}$$

Since the elements we are dealing with (both flat and circular arc lines) are of constant Jacobian with $J(\xi) = \frac{L}{2}$ we can write equations (9.5) and (9.5) as,

$$h^{LUT} = \frac{1}{4\pi}\int_{-1}^{+1}\frac{1}{R_m}\mathbf{N}^{\mathbf{T}}\mathrm{d}\xi \tag{9.7}$$

$$g^{LUT} = \frac{1}{4\pi}\int_{-1}^{+1}\ln\left[\frac{1}{R_m}\right]\mathbf{N}^{\mathbf{T}}\mathrm{d}\xi \tag{9.8}$$

From these it can be seen that the following adjustments need to be made before the values in the LUT can be used in equation (9.4).

$$h = h^{LUT} \tag{9.9}$$

$$g = \left(g^{LUT} - a\right) L \tag{9.10}$$

where h and g are matrix coefficients and,

$$a = \frac{1}{2\pi} \int_{-1}^{+1} \ln(L)\, \mathbf{N}^{\mathbf{T}} \frac{J(\xi)}{L} \mathrm{d}\xi \tag{9.11}$$

Moreover, we can follow the same procedure as chapter 6 and noting that all of the terms except for $\mathbf{N}^{\mathbf{T}}$ are constants

$$\begin{aligned} a &= \frac{1}{2\pi} \int_{-1}^{+1} \ln(L)\, \mathbf{N}^{\mathbf{T}} \frac{J(\xi)}{L} \mathrm{d}\xi \\ &= \text{const} \int_{-1}^{+1} \mathbf{N}^{\mathbf{T}} \mathrm{d}\xi \\ &= \alpha\beta \end{aligned} \tag{9.12}$$

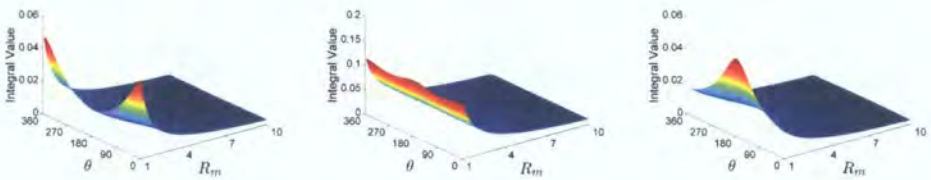
Consideration of the integral in equation (9.12) produces the well known coefficients for quadratic elements of

$$\alpha = \frac{\ln(L)}{24\pi} \tag{9.13}$$

$$\beta = \begin{pmatrix} 1 & 4 & 1 \end{pmatrix} \tag{9.14}$$

In the same way as was described for elastostatics, the LUTs in equations (9.7) and (9.8) can be used as a direct replacement for Gauss-Legendre integration of equation (9.4).

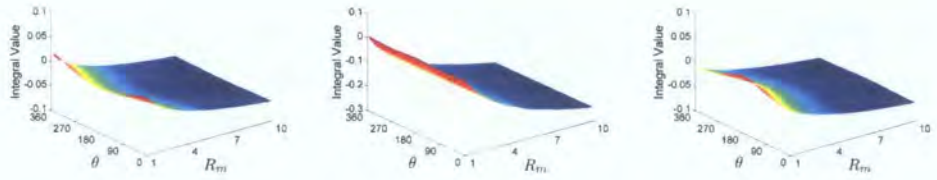
Figures 9.1 and 9.2 display surface plots for the integrals. It can be seen that



(a) End-node 1 (b) Mid-node (c) End-node 2

Figure 9.1: Surface plots of h

the functions are smoothly varying. As a result it is possible to fit basis functions to the surfaces to allow the rapid computation of the integrals.



(a) End-node 1 (b) Mid-node (c) End-node 2

Figure 9.2: Surface plots of g

Equations 9.15 to 9.20 show the surface fit equations for h and g for $2 \leq R_m \leq 3$ subscripts designate the appropriate node. The associated error with each fit is less than 0.1%.

$$h_1 = \left[\begin{array}{l} 2.658R_m^{-1} + R_m^{-2}\left(1.340\sin\left(\theta\right) - 0.061\cos\left(2\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.163 - 0.126\sin\left(3\theta\right) - 0.367\cos\left(2\theta\right) \\ +0.018\cos\left(4\theta\right) \end{array} \right) \end{array} \right] \times 10^2 \tag{9.15}$$

$$h_2 = \left[R_m^{-1}\left(1.059 + 0.007\cos\left(2\theta\right)\right) + R_m^{-2}\left(0.011 - 0.034\cos\left(2\theta\right)\right)\right] \times 10 \tag{9.16}$$

$$h_3 = \left[\begin{array}{l} 2.658R_m^{-1} - R_m^{-2}\left(1.340\sin\left(\theta\right) + 0.061\cos\left(2\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.163 + 0.126\sin\left(3\theta\right) - 0.367\cos\left(2\theta\right) \\ +0.018\cos\left(4\theta\right) \end{array} \right) \end{array} \right] \times 10^2 \tag{9.17}$$

$$g_1 = \left[\begin{array}{l} -2.653\ln R_m + 1.326R_m^{-1}\sin\left(\theta\right) - 0.199R_m^{-2}\cos\left(2\theta\right) \\ -0.066R_m^{-3}\sin\left(3\theta\right) \end{array} \right] \times 10^2 \tag{9.18}$$

$$g_2 = \left[-1.061\ln R_m - 0.027R_m^{-1}\cos\left(2\theta\right)\right] \times 10 \tag{9.19}$$

$$g_3 = \left[\begin{array}{l} -2.653\ln R_m - 1.326R_m^{-1}\sin\left(\theta\right) - 0.199R_m^{-2}\cos\left(2\theta\right) \\ +0.066R_m^{-3}\sin\left(3\theta\right) \end{array} \right] \times 10^2 \tag{9.20}$$

Similarly to the stress analysis case the end-node surfaces can be calculated in an optimised manner due to the simplification of

$$h_1 = A + B \tag{9.21}$$

$$h_3 = A - B \tag{9.22}$$

and

$$g_1 = C + D \tag{9.23}$$

$$g_3 = C - D \tag{9.24}$$

Considering figures 9.1 and 9.2 it should be noted that the gradient for the range $1 \leq R_m \leq 3$ is much higher than for terms with higher R_m values. Equation 9.25 shows the surface fit for h_2 with $3 \leq R_m \leq 10$,

$$h_2 = \left[1.062R_m^{-1} - 0.040R_m^{-3}\cos(2\theta)\right] \times 10 \tag{9.25}$$

Moreover further reduction in the interval considered in the R_m direction either increases the accuracy of the surface fit or reduces the number of terms required to fit accurately over the dataset. For example, equation (9.26) shows the h_2 surface fit for the interval $5 \leq R_m \leq 10$ and equation (9.27) shows the same for $9 \leq R_m \leq 10$.

$$h_2 = \left[1.061R_m^{-1} - 0.230R_m^{-4}\cos(2\theta)\right] \times 10 \tag{9.26}$$

$$h_2 = \left[1.061R_m^{-1}\right] \times 10 \tag{9.27}$$

This emphasises the importance of choosing a suitable strategy for splitting the R_m direction, to reduce the computational cost at run-time whilst ensuring that the code is easy to maintain and implement. More research is warranted in this area.

9.1.2 Acoustics

Acoustic analysis for boundary integral formulations has been presented in chapter 3. The boundary integral equation for acoustics is given by,

$$\phi(\kappa) + \int_\Gamma \frac{\partial G(\kappa, Q)}{\partial n}\phi(Q)\,\mathrm{d}\Gamma(Q) = \int_\Gamma G(\kappa, Q)\frac{\partial \phi(Q)}{\partial n}\mathrm{d}\Gamma(Q) + \phi_i \tag{9.28}$$

where ϕ is the acoustic pressure and is complex since it contains both magnitude and phase components, ϕ_i is the incident wave and G and $\frac{\partial G}{\partial n}$ are the fundamental solutions given by

$$G = \frac{i}{4}H_0(kr) \tag{9.29}$$

$$\frac{\partial G}{\partial n} = \frac{k}{4}\frac{\partial r}{\partial n}\left[Y_1(kr) - iJ_1(kr)\right] \tag{9.30}$$

where H_0 is a Hankel function of the first kind given by,

$$H_0 = J_0 + iY_0 \tag{9.31}$$

J_n and Y_n are Bessel functions of the first and second kind respectively.

Following a similar procedure as for heat transfer problems,

$$\phi(\kappa) + \sum_{elem} \int_{-1}^{+1} \frac{\partial G}{\partial n} \mathbf{N}^{\mathbf{T}} J(\xi) \, \mathrm{d}\xi \boldsymbol{\phi} = \sum_{elem} \int_{-1}^{+1} G \mathbf{N}^{\mathbf{T}} J(\xi) \, \mathrm{d}\xi \frac{\boldsymbol{\partial \phi}}{\boldsymbol{\partial n}} + \phi_i \tag{9.32}$$

Thus it is necessary to create the following LUTs.

$$G^{LUT} = \int_{-1}^{+1} \frac{i}{4} H_0(kr) \mathbf{N}^{\mathbf{T}} J(\xi) \, \mathrm{d}\xi \tag{9.33}$$

$$\left(\frac{\partial G}{\partial n}\right)^{LUT} = \int_{-1}^{+1} \frac{k}{4} \frac{\partial r}{\partial n} \left[Y_1(kr) - iJ_1(kr)\right] \mathbf{N}^{\mathbf{T}} J(\xi) \, \mathrm{d}\xi \tag{9.34}$$

However, as a result of the Bessel functions within the fundamental solutions it is not possible to convert the LUT form into the required form for use within equation (9.32) without integrating a Bessel function in the adjustment.

One technique for potentially overcoming the problem of Bessel functions within the fundamental solutions is to approximate the Bessel function by a high order polynomial (Press, 2002). This technique has been applied by Honnor *et al.* (2007) for the rapid integration of acoustic problems using the partition of unity boundary element method (PUBEM). The ability to describe the Bessel functions in terms of a high order polynomial potentially allows the conversion of the fundamental solutions to a form that can be evaluated at low computational cost.

9.1.3 Three-dimensional analysis

Extension of the proposed integration techniques for 3-dimensional problems suffers from the associated increase in parameters required to describe a 3-dimensional boundary element.

For the initial assessment of 3-dimensional problems it seems necessary to place restrictions on the type of element. Figure 9.3 shows a rectangular element and the parameters necessary to define this particular case in space when integrating over this element and considering a given collocation point location. It can be seen that there are 8 main independent parameters that define the 3-dimensional quadrilateral case.

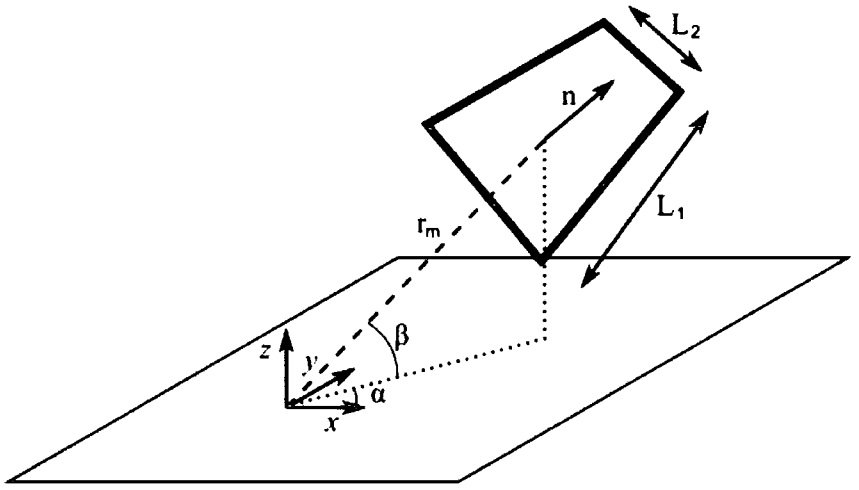


Figure 9.3: Three dimensional element with defining parameters

- α - Angle between x axis and r_m.
- β - Angle between xy plane and r_m.
- r_m - Distance between source point and field element midpoint.
- L_1 - Length of side 1 of the element.
- L_2 - Length of side 2 of the element.
- γ - Angular orientation of side 1 of the element.
- n_x - Element normal in the x coordinate direction.
- n_y - Element normal in the y coordinate direction.

These parameters can be reduced by consideration of an element scale parameter $\bar{L} = \frac{L_1}{L_2}$ and by rotating the element around the z axis such that $\alpha = 0°$. Thus the parameter set has been reduced from 8 to 6 parameters.

Table 9.1 shows the memory requirements for LUTs generated for these parameter sets assuming that a similar refinement is implemented for the LUTs in the 3-dimensional case as implemented in the two-dimensional scheme. No assumptions on symmetry have been included in the memory requirements although it is probable that in the angular directions symmetries will exist thus reducing the actual memory

	Memory Requirement
Non-Interpolated	9.93YB
Interpolated	11.71ZB

Table 9.1: Three-dimensional memory requirements - No symmetries

cost. A yottabyte (YB) is defined as 2^{80} bytes ($\sim 10^{12}$ terabytes), a zettabyte (ZB) is defined as 2^{70} bytes ($\sim 10^{9}$ terabytes) and an exabyte (EB) is defined as 2^{60} bytes ($\sim 10^{6}$ terabytes). If symmetries are assumed within the angular parameters such that only 180° need to be stored for corner nodes and only 90° need to be stored for mid-side nodes. These memory requirements are unreasonable for the foreseeable

	Memory Requirement
Non-Interpolated	337.48ZB
Interpolated	398.28EB

Table 9.2: Three-dimensional memory requirements - Symmetries

future and as a result the use of LUTs as a means to accelerate the integration phase for 3-dimensional problems is not possible. However, the use of surface fitting techniques could be applied to the 6 parameter space to produce equations for each of the required functions.

Difficulties arise within the use of surface fits in that they need to be computationally cheaper than the associated variable Gauss-Legendre quadrature. For the proposed 2 parameter fit for 2-dimensional problems they are 53% faster. The introduction of an additional 4 parameters could cause this computational cost to increase significantly. As a result careful analysis will be required in the preliminary stages to ensure that appropriate basis functions are chosen.

Another alternative is to place additional restrictions on the element. It is possible to ensure that elements are formed as squares by adapting the meshing routine to favour elements of this type. Additionally, a large number of elements in a typical analysis will be aligned with one of the coordinate planes. These assumptions reduce the number of parameters to 3 (figure 9.4).

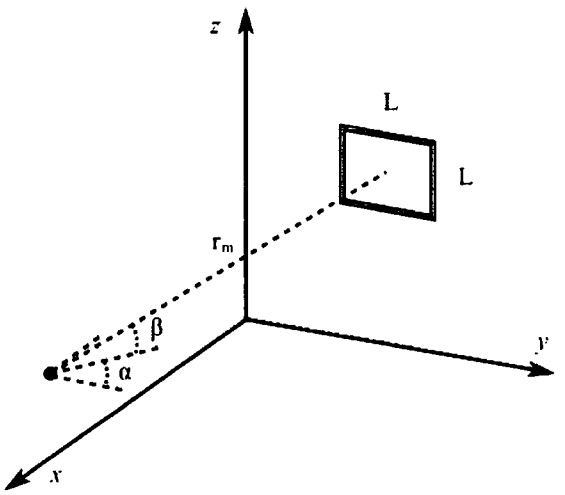


Figure 9.4: Alternative definition for three dimensional element

- α - Angle between y axis and r_m
- β - Angle between xy plane and r_m
- $R_m = \frac{r_m}{L}$

where r_m is the distance between the source point and the centre of the field element and L is the length of the element sides. The reduction in parameter count, although limiting of the number of cases that the technique can be applied, will ease initial assessment of the technique.

9.2 Equation solution

The use of a complete but approximate LU preconditioner as part of a GMRES iterative solver can be applied to any matrix problem of the form,

$$\mathbf{Ax} = \mathbf{b} \tag{9.35}$$

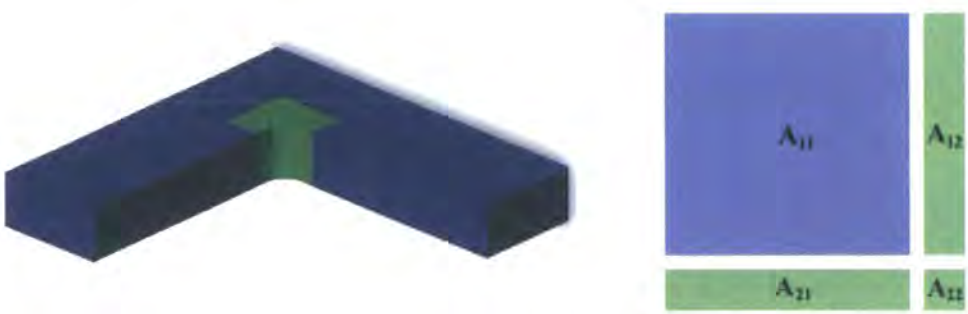
However, although this technique should be effective for a wide range of problems; as a result of the preconditioner being close to the inverse matrix, it may not be the optimum solution strategy. It is important to consider the specific attributes of the problem under consideration, for instance,

- Is the **A** matrix banded in any way?
- Is the **A** matrix a sparse matrix?
- Is the problem small?

If the matrix is banded or sparse an alternative preconditioner may be cheaper to apply than a complete LU factorisation as a result of the sparsity that can be achieved within the preconditioner. Moreover if the problem cannot be considered small then the storage and application of the preconditioner may be of concern.

Multi-zone problems feature block sparsity as a result of the interlinking nature between the sub-matrices within the global **A** matrix. It is possible to enclose particular features, which are likely to be perturbed in some manner, within individual zones. Thus, upon reanalysis only blocks associated with that zone need to be recomputed into the global **A** matrix. This is beneficial from a solution point of view because the matrix blocks can be ordered such that a large part of a triangular decomposition can be stored from the previous solution phase and hence, reduce the time taken to calculate the overall factorisation.

Figure 9.5 shows an example problem with zones indicated and the associated matrix configuration. The fillet zone is small in comparison with the main body



(a) Model (b) Matrix layout

Figure 9.5: Multi-zone problem

of the problem. Thus if the fillet is increased or reduced in size only the relatively small sections $\mathbf{A}_{12}$, $\mathbf{A}_{21}$ and $\mathbf{A}_{22}$ need to be recalculated. Moreover the previous factorisation of $\mathbf{A}_{11}$ can be reused ensuring a large saving in computational cost.

9.3 Dual boundary element method

The dual boundary element method (DBEM) developed by Portela and Aliabadi (1992) for 2-dimensional problems, and by Mi and Aliabadi (1992) for 3-dimensional problems, is a computationally efficient approach for the analysis of crack problems.

Consideration of co-planar crack surfaces allows the derivation of displacement and traction integral equations similar in form to those derived in chapter 3. The

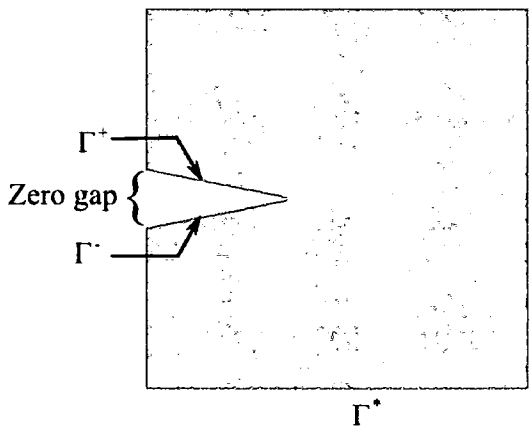


Figure 9.6: Co-planar crack surfaces (Aliabadi, 2002)

boundary of the problem is defined as $\Gamma_c^{\pm}$ referring to upper and lower crack surfaces and Γ^* being the rest of the boundary.

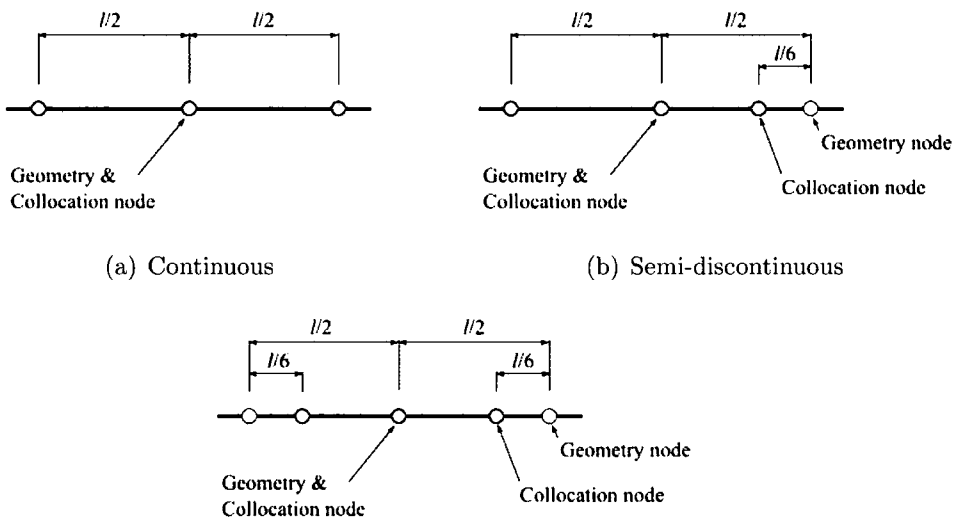
$$c_{ij}\left(x^+\right) u_j\left(x^+\right) + c_{ij}\left(x^-\right) u_j\left(x^-\right) = \int_\Gamma U_{ij}\left(x^+, x\right) t_j\left(x\right) \mathrm{d}\Gamma - \oint_\Gamma T_{ij}\left(x^+, x\right) u_j\left(x\right) \mathrm{d}\Gamma \quad (9.36)$$

$$\frac{1}{2} t_j\left(x^-\right) - \frac{1}{2} t_j\left(x^+\right) = n_i\left(x^-\right) \oint_\Gamma D_{kij}\left(x^-, x\right) t_k\left(x\right) \mathrm{d}\Gamma - n_i\left(x^-\right) \oint_\Gamma S_{kij}\left(x^-, x\right) u_k\left(x\right) \mathrm{d}\Gamma \quad (9.37)$$

The DBEM uses the displacement integral equation (equation (9.36)) to collocate on x^+ on the upper crack surface, Γ^+. Whereas the traction integral equation (equation (9.37)) is collocated on x^- on the lower crack surface, Γ^-.

Issues arise within the DBEM for the integration of the necessary Cauchy and Hadamard integrals in equation (9.37). Conditions assumed during the derivation

of the DBEM require certain conditions to be imposed on the shape functions for the crack surfaces used within the DBEM. It is necessary that the displacement components and the derivatives of the displacement components have continuity, the use of discontinuous elements along the crack surface fulfils this requirement. As continuous elements are employed around the remainder of the boundary it is necessary to have semi-discontinuous elements at the intersection between a crack and the edge, this prevents common nodes being placed at the intersection.



(a) Continuous

(b) Semi-discontinuous

(c) Discontinuous

Figure 9.7: Quadratic element types (Aliabadi, 2002)

The use of the proposed integration routines is possible for situations where R_m is in the appropriate range. It is necessary, however, to have separate datasets for continuous, semi-discontinuous and discontinuous elements because of the different shape functions required for each case. For situations where R_m is out of the data-set's bounds it will be necessary to perform the integrations using alternative techniques such as Gauss-Legendre quadrature possibly employing the techniques of Telles (1987) and Kutt (1975) to compensate for the singular nature of the integrals.

9.3.1 Crack growth

Crack growth within the DBEM can be accommodated by the addition of elements at the crack tip as it extends. The addition of these elements will cause the linear

system of equations to be increased, however, to allow a rapid re-solution to the problem it is possible to add the extra terms to the bottom corner of the original matrix problem, figure 9.8. Thus, it is possible to use the previous **L** and **U** factors and then complete the remaining part of the factorisation. It is important to ensure

Figure 9.8: Matrix for the reanalysis problem of crack growth

that an appropriate level of grading is implemented within the additional elements, and as a result for small increments in crack growth it may be necessary to merely increase the length of particular elements.

9.4 Optimisation

The acceleration of computations can be exploited for other situations, for example, in evolutionary stress optimisation. Algorithms such as those proposed by Cervera (2003) require an iterative procedure within the geometric structure, as a result the geometry undergoes a large number of small perturbations for which the use of a good preconditioner is essential. Cervera used non-uniform rational B-splines within the geometry definition and because of this it is not possible to employ the fast integration techniques proposed for the spline sections of geometry. However, the problem will still result in a linear system of equations.

The real-time analysis and dynamic update of contours that has been presented within this thesis changes the design paradigm that currently exists within the early stages of design for components. The rapid analysis allows a quick comparison of

multiple designs and the ability to allow the original design to be quickly driven to an optimum by aid of visual feedback from the dynamically updated contours.

The alteration to the design paradigm is paramount to this work and highlights the benefits of the proposed techniques. Mathematical optimisations suffer from the potential of having multiple minima within the solution space and as a result a mathematical optimisation can converge to a local minimum but not a global minimum. By utilising the experience of an engineer to help guide the design it is possible to avoid local minima and investigate the complete solution space quickly.

Moreover, the ease and speed of reanalysis allows inexperienced engineers to rapidly gain knowledge of complicated structures and the interactions that can occur between geometric features. Figure 9.9 identifies the interaction between two holes within a rectangular plate under uniaxial tension, with a real-time update of contours it is possible for the student engineer to vary the distance between the holes and see how the stress field is affected.

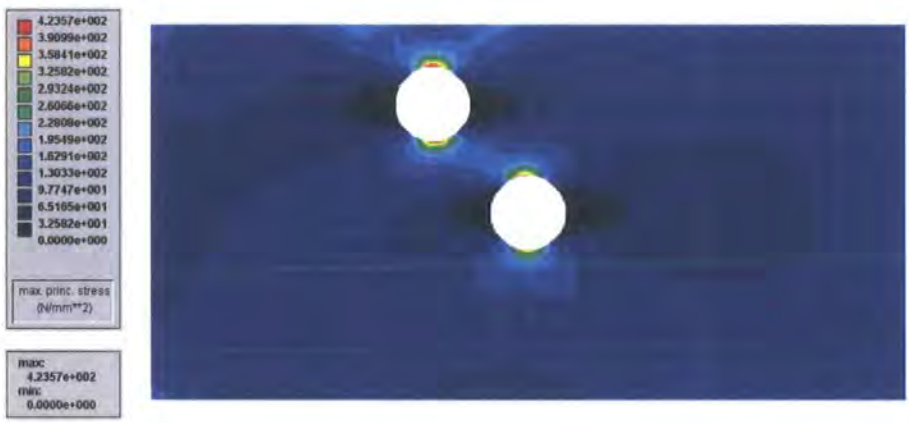


Figure 9.9: Interaction within the stress field between two holes in a rectangular plate

9.5 Concluding remarks

In this chapter a number of extensions to the current work have been examined with suggestions on techniques to implement the strategies. The use of LUTs and surface fits for alternative problems is possible assuming that the fundamental solutions

can be manipulated into the appropriate form without excessive cost of extracting particular values. For 3-dimensional problems it is necessary to make assumptions about the element shape (to reduce the overall parameter count). However, this results in 3 parameters and as a result it is not feasible to store LUTs for this type of problem. The use of surface fits could be the solution to this problem but a detailed analysis is necessary to determine the most efficient basis functions for the dataset.

The proposed technique for equation solution can be applied to a wide variety of problems and, as a result of the preconditioner being a good approximation to the inverse the solution, should be effective. However, specific attributes of the **A** matrix may make alternative preconditioners and solvers more efficient.

CHAPTER 10

Conclusions and recommendations for future work

In this thesis techniques have been presented for the acceleration of the analysis and in particular reanalysis of elastostatic problems modelled using the boundary element method. The problem has been tackled in two main parts. Consideration of the integration phase of the analysis and the solution of the resulting matrix equations. Each of the techniques has been presented and discussion of the main advantages and drawbacks of each method presented within the respective chapters. This final chapter summarises the main points of these conclusions and gives suggestions for future work.

10.1 Achievements

The main achievements of the current work can be stated as follows

- Implemented within the in-house Concept Analyst software a real-time analysis and dynamic updating of contours for both displacement and stress forms of the boundary integral equation.
- Profiled the current scheme for analysis and reanalysis to determine areas requiring improvement.

- Targeted two main areas for improvement: the integration routine and the solution of the matrix equation generated from the boundary element method.
- Integration can be accelerated by computing the integrals and storing them in some manner. These can then be extracted at run-time at a lower computational cost than equivalent Gauss-Legendre quadrature.
- Implemented look-up tables (LUTs) to store precomputed integral values for flat and circular arc elements.
- Coordinate transformations were developed to allow the use of LUTs for arbitrarily angled elements.
- The use of interpolation within the R_m parameter presented. The use of interpolation allows smaller size LUTs to be employed.
- The refinement of LUTs was investigated to determine the appropriate refinement level to meet error requirements. Suggested refinement is 0.05° in the angular direction and a geometric progression for the scaling direction, R_m. For interpolated LUTs a geometric progression factor of $s = 1.03$ is required and for non-interpolated LUTs $s = 1.001$.
- The total memory requirement for non-interpolated LUTs is approximately 3GB. Use of interpolated LUTs reduces this to only 102MB.
- The strategy proposed on current hardware is a combination of both the non-interpolated and interpolated LUTs with a memory requirement of 911MB. Implementation of this strategy allows a feasible memory requirement for current hardware whilst optimising the efficiency of the proposed technique.
- LUTs have been structured within memory to optimise the use of rapid caching of memory addresses, resulting in an approximately 75% increase in memory access times with repect to non-orientated LUTs. This seemingly trivial factor is essential to effective use of LUTs for this application.
- The use of LUTs produce savings of 47% for flat elements using non-interpolated LUTs and 38% for circular arc elements using interpolated LUTs

- Implemented least squares surface fits for integrals data for flat elements, orientated for fixed angles of ϕ.
- Two stage least squares surface fit routine employed to allow optimum fits to be found for particular error requirement. The initial stage reduces the basis functions to important terms for a particular boundary integral, while the second stage is a brute force approach to find the optimum fit.
- Investigated the use of a coordinate transformation to allow the application to arbitrary ϕ.
- The use of small look-up tables for values of $\sin(q\phi)$ and other basis functions, to accelerate the computation of trigonometric and logarithmic terms. Memory requirement for the surface fit basis LUTs is 5.8MB.
- It was found that savings of 74% are possible using surface fits for cases of fixed $\phi = 0°$, $90°$, $180°$ and $270°$. Employing coordinate transformations to allow the extension to arbitrary ϕ the time saving is reduced to 53% of an adaptive Gauss-Legendre quadrature scheme. However, the additional terms to which the proposed technique can now be applied increase the potential for computational saving.
- Investigated the use of preconditioners with a GMRES iterative solver.
- Eigenvalue distributions have been investigated that demonstrate the clustering ability of the proposed complete but approximate LU preconditioner.
- Multiple perturbations to the problem degrade the preconditioner and as a result an update strategy has been implemented in a secondary low priority thread.
- Effectiveness of the preconditioning strategy has been investigated for a variety of types and degree of perturbation.
- It was found that typical savings of between 75% and 85% are achieved in the equation solution stage with respect to a direct solver.

10.2 Conclusions

The techniques developed and presented in this thesis allow the solution in real-time and the dynamic update of contour plots for both two dimensional stress and displacement problems presented within the boundary element method. As this work is aimed at the reanalysis of problems and the real-time updating of contours the implementation within the boundary element method framework allows for the fast and reliable updating of the problem mesh after each perturbation. The techniques presented have been implemented within the in-house Concept Analyst software. This implementation has been performed in Visual C++. Approximately 7500 lines of code have been written within the Concept Analyst framework in the development of the LUT version and 15000 lines for the least squares surface fit version.

The techniques proposed within this thesis have been made based on the currently available hardware within the *average* engineering office. As a result it may be necessary to adjust the recommendations as hardware develops. For example, the use of non-interpolated LUTs requires a approximately 3GB of memory and as a result these cannot be currently implemented without the use of swap space and hence an additional computational cost. As technology improves and the *average* personal computer improves the use of non-interpolated LUTs will become feasible. It it prudent to note that within a large organisation the hardware that the *average* engineer employs is a costly asset to the organisation and as a result have a long shelf life with respect to the rate of change in technology. Margetts *et al.* (2005) employ specialised hardware and supercomputing power within their analysis, technology that potentially is not available within an organisation. It is the author's belief that LUTs should be re-investigated in the future with respect to implementation of non-interpolated LUTs.

Surface fits are the fastest technique for evaluating the boundary element integrals and as a result are the scheme recommended from this thesis. However, this speed of computation comes at the cost of accuracy, 0.1% of the integral value. As hardware develops surface fits will be consistently faster at evaluating the boundary element integrations as they are not affected by the increase in memory available on PCs. LUTs, however, although slower, will improve in accuracy as the memory

available on PCs increases. Thus if a higher degree of accuracy is required then the use of LUTs is recommended.

Splitting the main problem of accelerating the analysis and reanalysis of elastostatic problems into two sections has allowed the development of two separate but compatible techniques. As the techniques are distinct they have the ability to be developed separately for use within alternative areas. This modular behaviour is similar to the concept of object orientated programming, with the definition of interfaces such that modules can be switched as long as the interfaces match. This ability lends itself to the development for alternative applications such as potential flow because the matrices will be of a similar form to those found in elastostatics; diagonally dominant but fully populated; as a result it should only be necessary to alter the integration routine. In this way, existing boundary element codes can readily be modified to take advantage of the acceleration strategies developed in this work.

10.3 Recommendations for future work

The techniques implemented within this thesis have shown good performance. The aims of this PhD have also been accomplished with the production of a real-time elastostatic analysis package. However, whilst working on this topic a number of areas for future research have arisen; these areas have been discussed within chapter 9.

The extension of the least squares surface fits to circular arc elements would allow a larger proportion of elements to be integrated using the proposed technique. However, as has been noted for the LUTs the introduction of a circular arc elements complicates the integral data. As a result it is expected that the surface fit equations generated would be more complicated than those currently implemented so careful selection of basis functions will be required to optimise the surface fits.

Additionally, the extension to alternative types of problem such as potential flow shows great promise. The integrals can be transformed into a form suitable for LUTs and surface fitting techniques. More complicated forms of problems, such as

acoustics, have more complicated fundamental solutions and as a result conversion to a form suitable for implementing LUTs or surface fits may not be possible. However, the use of an eighth order polynomial as a representation for Bessel functions (Press, 2002) shows that there is potential for this technique.

The current implementation is aimed at the real-time analysis of problems in two dimensions. However, the use of similar techniques could be used to accelerate the solution of problems within three dimensions. An initial investigation scheme has been outlined, reducing the required number of parameters required to fully describe elements within a 3-dimensional model. The implementation of LUTs for three dimensions will require a large amount of memory and as a result it is the author's belief that these, using current hardware, are unfeasible. Hence, the use of a least squares surface fit will be the most successful line of research.

If the real-time reanalysis ideas proposed in this thesis are to be adopted for general mechanical design, it is important that they not be restricted, in the long term, to the small sized problems considered in this work. Extension to larger problems will occur naturally with further developments in computer hardware, as well as some of the ideas presented above. Careful analysis will have to be performed into the way errors in matrix terms propagate through the solution process to errors in the stress results that are the outcome of the analysis that will be used by engineers. Such an analysis has been performed in this work but a considerably more extensive programme should be carried out for more substantially sized problems.

It has been found that reanalysis times are dependent on the number of elements that are changed in a geometric design change as a result of the consequent remeshing. There is considerable scope within three dimensional reanalysis for research into techniques for meshing and remeshing that minimise the number of changed elements. This might involve, for example, automatic feature recognition and enclosing of important design features in small zones.

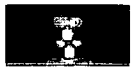
10.4 Summary

The need for rapid analysis within the early conceptual design stage is essential for maintaining the necessary competitive edge to succeed. Changing the design paradigm through real-time analysis has caused the landscape for the conceptual engineer to be changed completely. The ability to rapidly *experiment* with a design allows a more fluid development cycle, essential within the formative stages of a product.

We will always be asked to push the boundaries to the problems that we face and only by continually testing and probing will we know where these boundaries lie. It is only through the innovative use of technology that engineers will be able to maintain the competitive advantage required to succeed. As a result the learning process never ends, however far we progress we can always move forward.

Ancora imparo

- *Michelangelo Buonarroti*



Bibliography

M. Abramowitz and I. A. Stegun. *Handbook of mathematical functions with formulas, graphs and mathematical tables.* Dover Publications, New York, 9th edition, 2002.

A. I. Abreu, W. J. Mansur, and J. A. M. Carrer. Initial conditions contribution in a BEM formulation based on the convolution quadrature method. *Int. J. for Num. Meths. in Eng.*, 67:417–434, 2006.

F. Ahmad. A system of equations with a tridiagonal coefficient matrix. *Applied Mathematics and Computation*, 159:435–438, 2004.

M. H. Aliabadi. *The boundary element method: Applications in solids and structures*, volume 2. John Wiley & Sons Inc., 2002.

W. E. Arnoldi. The principle of minimized iterations in the solution of the matrix eigenvalue problem. *Quarterly of Applied Mathematics*, 9:17–29, 1951.

S. N. Atluri and T. Zhu. A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Comput. Mech.*, 22:117–127, 1998.

C. E. Augarde and A. J. Deeks. On the effects of nodal distributions for imposition of essential boundary conditions in the MLPG meshfree method. *Comm. in Num. Meths. in Eng.*, 21:389–395, 2005.

O. Axelsson. *Iterative solution methods.* Cambridge University Press, 1994.

E. Babolian, M. MasjedJamei, and M. R. Eslahchi. On numerical improvement of Gauss-Legendre rules. *Applied Mathematics and Computation*, 160:779–789, 2005.

H. Bae, R. V. Grandhi, and R. A. Canfield. Accelerated engineering design optimization using successive matrix inversion method. *Int. J. for Num. Meths. in Eng.*, 66:1361–1377, 2006.

J. Baglama, D. Calvetti, G. H. Golub, and L. Reichel. Adaptively preconditioned GMRES algorithms. *SIAM J. Sci. Comput.*, 20(1):243–269, 1998.

R. Beauwens. Iterative solution methods. *Applied Num. Mathematics*, 51:437–450, 2004.

A. A. Becker. *The boundary element method in engineering: A complete course.* McGraw Hill, London, 1992.

T. Belytschko, Y. Y. Lu, and L. Gu. Element-free Galerkin methods. *Int. J. for Num. Meths. in Eng.*, 37:229–256, 1994.

M. Bollhöfer. A robust and efficient ILU that incorporates the growth of the inverse triangular factors. *SIAM J. Sci. Comput.*, 25(1):86–103, 2003.

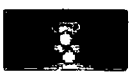
M. Bollhöfer and V. Mehrmann. Algebraic multilevel methods and sparse approximate inverses. *SIAM J. Matrix Anal. Appl.*, 24(1):191–218, 2002.

C. A. Brebbia. *The boundary element method for engineers.* Pentech Press, London, 1978.

C. A. Brebbia and J. Dominguez. *Boundary elements - An Introductory Course.* Computational Mechanics Publications, Southampton, 1989.

C. A. Brebbia and S. Walker. *Boundary element techniques in engineering.* Butterworths & Co. (Publishers) Ltd., 1980.

S. Bu and T. G. Davies. Effective evaluation of non-singular integrals in 3D BEM. *Advances in Engineering Software*, 23:121–128, 1995.



Z. Cao and X. Yu. A note on weighted FOM and GMRES for solving nonsymmetric linear systems. *Applied Mathematics and Computation*, 151:719–727, 2004.

B. Carpentieri, I. S. Duff, and L. Giraud. Robust preconditioning of dense problems from electromagnetics. In L. Vulkov, J. Waśniewski, and P. Yalamov, editors, *Numerical Analysis and Its Applications. Lecture Notes in Computer Science 1988*, pages 170–178. Springer, 2000.

E. Cervera. *Evolutionary structural optimisation based on boundary element representation of B-spline geometry.* PhD thesis, University of Durham, 2003.

E. Cervera and J. Trevelyan. Evolutionary structural optimisation based on boundary representation of NURBS. Part I: 2D algorithms. *Computers and Structures*, 83:1902–1916, 2005a.

E. Cervera and J. Trevelyan. Evolutionary structural optimisation based on boundary representation of NURBS. Part II: 3D algorithms. *Computers and Structures*, 83:1917–1929, 2005b.

S. C. Chapra and R. P. Canale. *Numerical methods for engineers.* McGraw Hill, 4th edition, 2002.

K. Chen. Efficient iterative solution of linear systems from discretizing singular integral equations. *Electronic Trans. on Num. Anal.*, 2:76–91, 1994.

K. Chen. On a class of preconditioning methods for dense linear systems from boundary elements. *SIAM J. Sci. Comput.*, 20(2):684–698, 1998.

K. Chen and P. J. Harris. Efficient preconditioners for iterative solution of the boundary element equations for the three-dimensional Helmholtz equation. *Applied Num. Mathematics*, 36:475–489, 2001.

Z-S. Chen and H. Waubke. A Burton-Miller collocation formulation of FMM for acoustic problems. In Z. H. Yao, M. W. Yuan, and W. X. Zhong, editors, *Computational Mechanics: Abstract (Volume 1)*, page 216. WCCM VI in conjunction with APCOM '04, Tsinghua University Press and Springer-Verlag, September 2004.

T. A. Cruse. Numerical solutions in three dimensional elastostatics. *Int. J. Solids Structures*, 5:1259–1274, 1969.

T. A. Cruse. Application of the boundary-integral equation method to three dimensional stress analysis. *Computers and Structures*, 3:509–527, 1973.

T. A. Cruse. An improved boundary-integral equation method for three dimensional elastic stress analysis. *Computers and Structures*, 4:741–754, 1974.

T. A. Cruse and F. J. Rizzo. A direct formulation and numerical solution of the general transient elastodynamic problem. I. *J. of Math. Anal. and Apps.*, 22: 244–259, 1968.

T. A. Cruse and W. Vanburen. Three-dimensional elastic stress analysis of a fracture specimen with an edge crack. *Int. J. Fracture Mechanics*, 7(1):1–15, 1971.

P. J. Davis and P. Rabinowitz. *Methods of numerical integration.* Academic Press, Orlando, 2 edition, 1984.

A. J. Deeks. Semi-analytical analysis of two-dimensional domains with similar boundaries. *Structural Engineering and Mechanics*, 14:99–118, 2002.

A. J. Deeks. Scaled boundary methods: Advantages for elastostatics. In Z. H. Yao, M. W. Yuan, and W. X. Zhong, editors, *Computational Mechanics*, pages 288–293. WCCM VI in conjunction with APCOM '04, Tsinghua University Press and Springer-Verlag, September 2004.

A. J. Deeks and J. P. Wolf. A virtual work derivation of the scaled boundary finite-element method for elastostatics. *Comput. Mech.*, 28:489–504, 2002.

L. M. Delves and J. L. Mohamed. *Computational methods for integral equations.* Cambridge University Press, Cambridge, 1985.

I. S. Duff. The impact of high-performance computing in the solution of linear systems: Trends and problems. *J. of Comp. and App. Mathematics*, 123:515–530, 2000.

I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct methods for sparse matrices.* Clarendon Press, Oxford, 1986.

U. Eberwien, C. Duenser, and W. Moser. Efficient calculation of internal results in 2D elasticity BEM. *Eng. Anal. with Boundary Elements*, 29:447–453, 2005.

Y. A. Erlangga, C. Vuik, and C. W. Oosterlee. On a class of preconditioners for solving the Helmholtz equation. *Applied Num. Mathematics*, 50:409–425, 2004.

R. T. Fenner. *Finite element methods for engineers.* MacMillan, London, 1975.

C. A. J. Fletcher. *Computational techniques for fluid dynamics*, volume 1. Springer-Verlag, Berlin, 1988.

J. R. Gilbert and S. Toledo. An assessment of Incomplete-LU preconditioners for nonsymmetric linear systems. *Informatica*, 24:409–425, 2000.

P. González, T. F. Pena, J. C. Cabaleiro, and F. F. Rivera. Dual BEM for crack growth analysis on distributed-memory multiprocessors. *Advances in Engineering Software*, 31:921–927, 2000.

P. González, T. F. Pena, and J. C. Cabaleiro. Parallel sparse approximate preconditioners applied to the solution of BEM systems. *Eng. Anal. with Boundary Elements*, 28:1061–1068, 2004.

A. Greenbaum. Comparison of splittings used with the conjugate gradient algorithm. *Numerische Mathematik*, 33:181–194, 1979.

A. Greenbaum. *Iterative methods for solving linear systems.* Society for Industrial and Applied Mathematics, Philadelphia, 1997.

A. El Guennouni, K. Jbilou, and H. Sadok. The block Lanczos method for linear systems with multiple right-hand sides. *Applied Mathematics and Computation*, 51:243–256, 2004.

M. H. Gutknecht. Variants of BiCGStab for matrices with complex spectrum. *SIAM J. Sci. Comput.*, 14(5):1020–1033, 1993.

W. Hackbusch. *Iterative solution of large sparse systems of equations.* Springer-Verlag, 1994.

R. W. Haywood. *Thermodynamic tables in SI (metric) units.* Cambridge University Press, 1998.

M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. of Res. of the National Bureau of Standards*, 49(6):409–436, 1952.

M.E . Honnor, J. Trevelyan, and D. Huybrechs. Numerical evaluation of 2D partition of unity boundary integrals for Helmholtz problems. *Comput. Methods Appl. Mech. Eng.*, 2007. Submitted.

S. R. Idelsohn and E. Oñate. To mesh or not to mesh. that is the question... *Comput. Methods Appl. Mech. Eng.*, 195:4681–4696, 2006.

N. I. Ioakimidis. On the Gaussian quadrature rule for finite-part integrals with a first-order singularity. *Comm. in Num. Meths. in Eng.*, 2:123–132, 1986.

M. A. Jaswon. Integral equation methods in potential theory - I. *Proc. Royal Society London*, A275:23–32, 1963.

M. A. Jaswon and G. T. Symm. *Integral equation methods in potential theory and elastostatics.* Academic Press, London and New York, 1977.

L. Jun, G. Beer, and J. L. Meek. Efficient evaluation of integrals of order $\frac{1}{r}$, $\frac{1}{r^2}$, $\frac{1}{r^3}$ using Gauss quadrature. *Eng. Anal. with Boundary Elements*, 2(3):118–123, 1985.

J. H. Kane. *Boundary element analysis in engineering continuum mechanics.* Prentice Hall, 1994.

J. H. Kane, A. Gupta, and S. Saigal. Reusable intrinsic sample point (RISP) algorithm for the efficient numerical integration of three dimensional curved boundary elements. *Int. J. for Num. Meths. in Eng.*, 28:1661–1676, 1989.

J. H. Kane, B. L. Keshava Kumar, and R. H. Gallagher. Boundary element iterative reanalysis for continuum structures. *J. of Engineering Mechanics*, 116(10):2293–2309, 1990.

U. Kirsch and G. Toledano. Approximate reanalysis for modifications of structural geometry. *Computers and Structures*, 16(1):269–277, 1983.

E. Kreyszig. *Advanced engineering mathematics.* Wiley, 8th edition, 1999.

A. R. Krommer and C. W. Ueberhuber. *Computational integration.* Society for Industrial and Applied Mathematics, Philadelphia, 1998.

H. R. Kutt. The numerical evaluation of principal value integrals by finite-part integration. *Numerische Mathematik*, 24:205–210, 1975.

J. C. Lachat and J. O. Watson. Effective numerical treatment of boundary integral equations: a formulation for three-dimensional elastostatics. *Int. J. for Num. Meths. in Eng.*, 10:991–1005, 1976.

L. Leu. Shape optimization by the boundary element method with a reduced basis reanalysis technique. *Structural Engineering and Mechanics*, 8(1):73–84, 1999.

C. Y. Leung and S. P. Walker. Iterative solution of large 3D BEM elastostatic analyses using the GMRES technique. *Int. J. for Num. Meths. in Eng.*, 40:2227–2236, 1997.

R. W. Lewis, K. Morgan, and O. C. Zienkiewicz. *Numerical methods in heat transfer.* John Wiley & Sons Inc., Chichester, 1981.

R. I. Mackie. An object-orientated approach to fully interactive finite element software. *Advances in Engineering Software*, 29(2):139–149, 1998.

S. Marburg and S. Schneider. Performance of iterative solvers for acoustic problems. Part 1: Solvers and effect of diagonal preconditioning. *Eng. Anal. with Boundary Elements*, 27:727–750, 2003.

R. J. Marczak. An object-orientated programming framework for boundary integral equation methods. *Computers and Structures*, 82:1237–1257, 2004.

R. J. Marczak. Object-orientated numerical integration - a template scheme for FEM and BEM applications. *Advances in Engineering Software*, 37:172–183, 2006.

L. Margetts, R. Ford, and C. Smethurst. Interactive finite element analysis. In *NAFEMS World Congress*, pages 1–12. NAFEMS World Congress, NAFEMS, May 2005.

J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Mathematics of Computation*, 31(137):148–162, 1977.

M. Merkel, V. Bulgakov, R. Bialecki, and G. Kuhn. Iterative solution of large-scale 3D BEM industrial problems. *Eng. Anal. with Boundary Elements*, 22:183–197, 1998.

Y. Mi and M. H. Aliabadi. Dual boundary element method for three-dimensional fracture mechanics analysis. *Engineering Analysis*, 10(2):161–171, 1992.

R. C. Mittal and A. H. Al-Kurdi. An efficient method for constructing an ILU preconditioner for solving large sparse nonsymmetric linear systems by the GMRES method. *Computers and Mathematics with Applications*, 45:1757–1772, 2003.

R. Morris. Acceleration of the integration phase of the boundary element method. Master's thesis, University of Durham, 2004.

K. W. Morton and D. F. Mayers. *Numerical solution of partial differential equations.* Cambridge University Press, Cambridge, 2005.

S. Mukherjee. *Boundary element methods in creep and fracture.* Applied Science Publishers, London and New York, 1982.

NAFEMS. *A finite element primer.* NAFEMS, Glasgow, 1992a.

NAFEMS. *A finite element dynamics primer.* NAFEMS, Glasgow, 1992b.

D. Nardini and C. A. Brebbia. A new approach to free vibration analysis using boundary elements. *Applied Mathematical Modeling*, 7(3):157–162, 1983.

NASA. Table of $\cos(a)$, 2006a. URL `http://www.grc.nasa.gov/WWW/K-12/airplane/tablcos.html`.

NASA. Table of $\sin(a)$, 2006b. URL `http://www.grc.nasa.gov/WWW/K-12/airplane/tablsin.html`.

H. Niki, K. Harada, M. Morimoto, and M. Sakakihara. The survey of preconditioners used in accelerating the rate of convergence in the Gauss-Seidel method. *J. of Comp. and App. Mathematics*, 164–165:587–600, 2004.

N. S. Ottosen and H. Petersson. *Introduction to the finite element method.* Pearson Education Ltd., 1992.

E. Perrey-Debain, O. Laghrouche, P. Bettess, and J. Trevelyan. Plane wave basis finite elements and boundary elements for three dimensional wave scattering. *Phil. Trans. Royal Society London A*, 362(1816):561–577, 2004.

A. Portela and M. H. Aliabadi. The dual boundary element method: Effective implementation for crack problems. *Int. J. for Num. Meths. in Eng.*, 33:1269–1287, 1992.

A. Portela, M. H. Aliabadi, and D. P. Rooke. Efficient boundary element analysis of sharp notched plates. *Int. J. for Num. Meths. in Eng.*, 32:445–470, 1991.

K. G. Pozrikidis. *Numerical computation in science and engineering.* Oxford University Press, 1998.

K. G. Prasad, J. H. Kane, D. E. Keyes, and C. Balakrishna. Preconditioned Krylov solvers for BEA. *Int. J. for Num. Meths. in Eng.*, 37:1651–1672, 1994.

W. H. Press. *Numerical recipes in C++: The art of scientific computing.* Cambridge University Press, 2002.

A. Ramage. An introduction to iterative solvers. In R. Crouch, editor, *Mathematics for engineers: The nonlinear deformation of solids*, EPSRC Summer School. Durham University, September 2006.

J. J. Rencis and K. C. Mann. The effect of essential boundary conditions on the convergence of iterative equation solvers in BEM. *Boundary Elements Communications*, 9:11–13, 1997.

J. A. Rice. *Mathematical statistics and data analysis.* Duxbury Press, 2nd edition, 1995.

F. J. Rizzo. An integral equation approach to boundary value problems of classical elastostatics. *Quarterly of Applied Mathematics*, 25:83–95, 1967.

Y. Saad. ILUT: a dual threshold incomplete LU factorization. *Numerical Linear Algebra with Applications*, 1(4):387–402, 1994.

Y. Saad. *Iterative methods for sparse linear systems.* PWS Publishing, Boston, 1996.

Y. Saad and M. Schultz. GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

H. A. Schenck. Improved integral formulation for acoustic radiation problems. *Journal of the acoustical society of America*, 44(1):41–58, 1968.

E. Schmidt. Zur theorie der linearen und nichtlinearen integralgleichungen. *Mathematische Annalen*, 63:433–476, 1907.

S. Schneider and S. Marburg. Performance of iterative solvers for acoustic problems. Part 2: Acceleration by ILU-type preconditioner. *Eng. Anal. with Boundary Elements*, 27:751–757, 2003.

H. R. Schwarz. *Finite element methods.* Academic Press, London, 1988.

G. L. G. Sleijpen and D. R. Fokkema. BiCGStab(l) for linear equations involving unsymmetric matrices with complex spectrum. *Electronic Trans. on Num. Anal.*, 1:11–32, 1993.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 17:140–148, 1885a.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 17:272–276, 1885b.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 18:91–96, 1885c.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 18:161–166, 1885d.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 19:84–90, 1886a.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 19:278–282, 1886b.

C. Somigliana. Sopra l'equilibrio di un corpo elastico isotropo. *Il Nuovo Cimento (Serie 3)*, 20:181–185, 1886c.

A. J. M. Spencer. *Continuum Mechanics.* Dover Publications, 2004.

K. A. Stroud. *Engineering Mathematics.* MacMillan, 4th edition, 1995.

E. Süli and D. F. Mayers. *An introduction to numerical analysis.* Cambridge University Press, Cambridge, 2003.

G. T. Symm. Integral equation methods in potential theory - II. *Proc. Royal Society London*, A275:33–46, 1963.

T. Takahashi, A. Kawai, and T. Ebisuzaki. Accelerating boundary integral equation method using a special-purpose computer. *Int. J. for Num. Meths. in Eng.*, 66: 529–548, 2006.

J. C. F. Telles. A self-adaptive coordinate transformation for efficient numerical evaluation of general boundary element integrals. *Int. J. for Num. Meths. in Eng.*, 24:959–973, 1987.

S. S. Terdalkar. *Graphically driven interactive stress reanalysis for machine elements in the early design stage.* PhD thesis, Worcester Polytechnic Institute, 2003.

S. S. Terdalkar and J. J. Rencis. Graphically driven interactive finite element stress reanalysis for machine elements in the early design stage. *Finite Elements in Analysis and Design*, 42:884–899, 2006.

The MathWorks Inc. *MatLab manual*, 2006.

E. E. Theotokoglou and G. Tsamasphyros. A modified gauss quadrature formula with special integration points for evaluation of quasi-singular integrals. *Eng. Anal. with Boundary Elements*, 30:758–766, 2006.

W. Thomson. On a mechanical representation of electric, magnetic and galvanic forces. *Cambridge and Dublin Math. J.*, 2:61–65, 1847. Reprinted in Math. and Phys. Papers, 76–80.

W. Thomson. Note on the integration of the equations of equilibrium of an elastic solid. *Cambridge and Dublin Math. J.*, 3:87–89, 1848. Reprinted in Math. and Phys. Papers, 97–99.

S. Timoshenko. *Theory of elasticity.* McGraw Hill, New York, 1 edition, 1934.

M. Tournour, J-P. Rossion, L. Bricteux, and C. McCulloch. Getting useful FEM and BEM vibro-acoustic solutions faster, using new solution methodologies. Technical report, LMS International, 2001.

J. Trevelyan. Concept Analyst 1.6. Concept Analyst Ltd., 2003.

J. Trevelyan. Private communication. Plane wave basis results. Results from application of the plane wave boundary element method, 2006.

J. Trevelyan. *Boundary Elements for Engineers: Theory and Application.* Computational Mechanics Publications, Southampton, 1994.

J. Trevelyan and P. Wang. Interactive re-analysis in mechanical design evolution. Part 1: Background and implementation. *Computers and Structures*, 79:929–938, 2001a.

J. Trevelyan and P. Wang. Interactive re-analysis in mechanical design evolution. Part 2: Rapid evaluation of boundary element integrals. *Computers and Structures*, 79:939–951, 2001b.

J. Trevelyan, D. J. Scales, R. Morris, and G. E. Bird. Acceleration of boundary element computations in reanalysis of problems in elasticity. In Z. H. Yao, M. W. Yuan, and W. X. Zhong, editors, *Computational Mechanics: Abstract (Volume 2)*, page 44. WCCM VI in conjunction with APCOM '04, Tsinghua University Press and Springer-Verlag, September 2004.

G. Tsamasphyros and E. E. Theotokoglou. A quadrature formula for integrals with nearby singularities. *Int. J. for Num. Meths. in Eng.*, 67:1082–1093, 2006.

F. P. Valente and H. L. G. Pina. Iterative solvers for BEM algebraic systems of equations. *Eng. Anal. with Boundary Elements*, 22:117–124, 1998.

A. van der Ploeg. Reordering strategies and LU-decomposition of block tridiagonal matrices for parallel processing. Technical report, Centrum voor Wiskunde en Informatica, 1996.

H. A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 13(2):631–644, March 1992.

L. Wang and A. Semlyen. Application of sparse eigenvalue techniques to the small signal stability analysis of large power systems. *IEEE Trans. Power Systems*, 5: 635–642, 1990.

J. R. Westlake. *A Handbook of Numerical Matrix Inversion and Solution of Linear Equations.* John Wiley & Sons Inc., New York, 1968.

J. P. Wolf and C. Song. The scaled boundary finite-element method - a primer: derivations. *Computers and Structures*, 78:191–210, 2000.

J. P. Wolf and C. Song. *Finite-element modelling of unbounded media.* John Wiley & Sons Inc., Chichester, 1996.

T. W. Wu. *Boundary element acoustics: Fundamentals and computer codes.* WIT Press, Southampton, 2000.

M. J. Young. *Mastering Visual C++ 6.* Sybex, Alameda, 1998.

W. Yu, Z. Wang, and X. Hong. Preconditioned multi-zone boundary element analysis for fast 3D electric simulation. *Eng. Anal. with Boundary Elements*, 28: 1035–1044, 2004.

X. Zhang and X. Zhang. Exact integrations of two-dimensional high-order discontinuous boundary elements of elastostatics problems. *Eng. Anal. with Boundary Elements*, 28:725–732, 2004a.

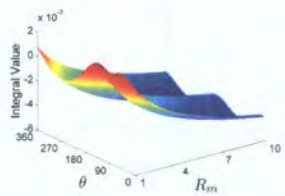
X. Zhang and X. Zhang. Exact integration for stress evaluation in the boundary element analysis of two-dimensional elastostatics. *Eng. Anal. with Boundary Elements*, 28:997–1004, 2004b.

O. C. Zienkiewicz and G. S. Holister. *Stress analysis: Recent developments in numerical and experimental methods.* John Wiley & Sons Inc., New York, 1965.

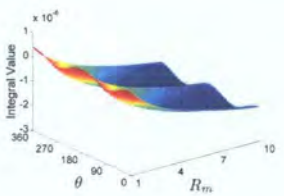
O. C. Zienkiewicz and R. L. Taylor. *The finite element method. Vol. 1. Basic formulation and linear problems*, volume 1. McGraw Hill, New York, 1989.

APPENDIX A

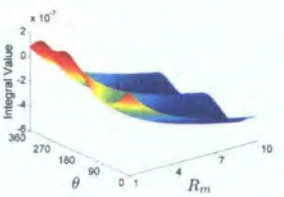
Surface Plots - g terms



(a) End-node 1

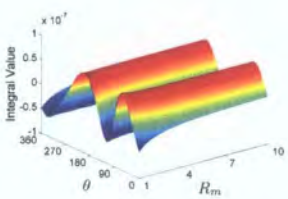


(b) Mid-node

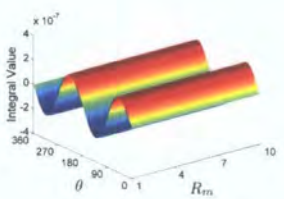


(c) End-node 2

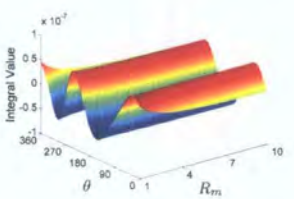
Figure A.1: $g_{\eta\eta}$



(a) End-node 1

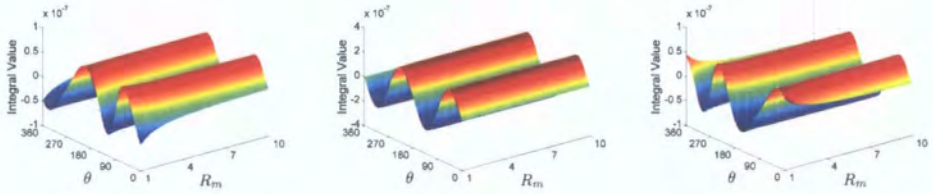


(b) Mid-node



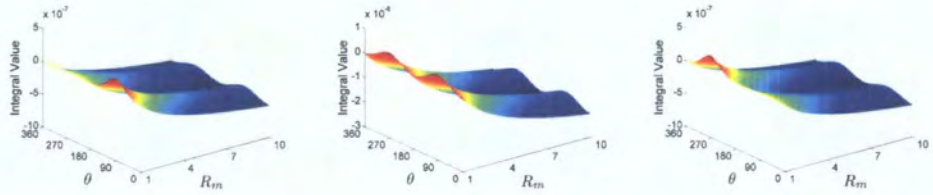
(c) End-node 2

Figure A.2: $g_{\eta\zeta}$ which is identical to $g_{\zeta\eta}$



(a) End-node 1 (b) Mid-node (c) End-node 2

Figure A.3: $g_{\zeta\eta}$

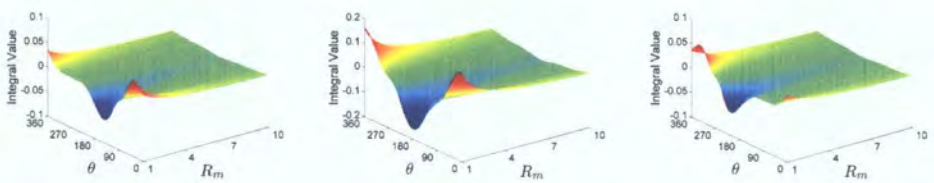


(a) End-node 1 (b) Mid-node (c) End-node 2

Figure A.4: $g_{\zeta\zeta}$

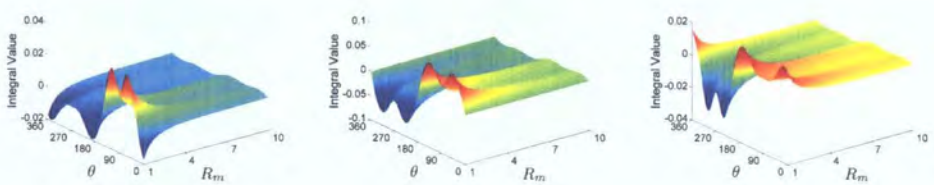
APPENDIX B

Surface Plots - h terms



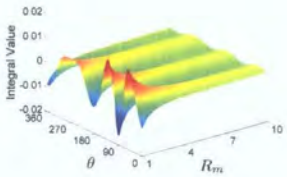
(a) End-node 1 (b) Mid-node (c) End-node 2

Figure B.1: $h_{\eta\eta}$

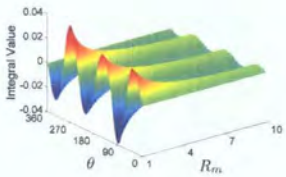


(a) End-node 1 (b) Mid-node (c) End-node 2

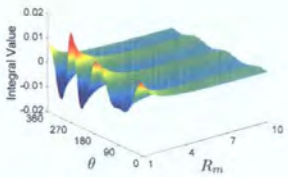
Figure B.2: $h_{\eta\zeta}$



(a) End-node 1

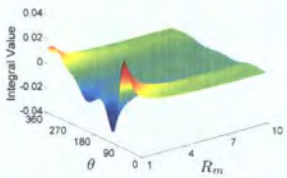


(b) Mid-node

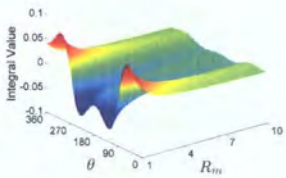


(c) End-node 2

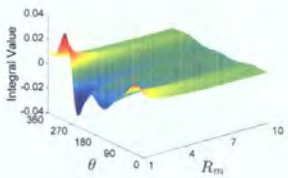
Figure B.3: $h_{\zeta\eta}$



(a) End-node 1



(b) Mid-node

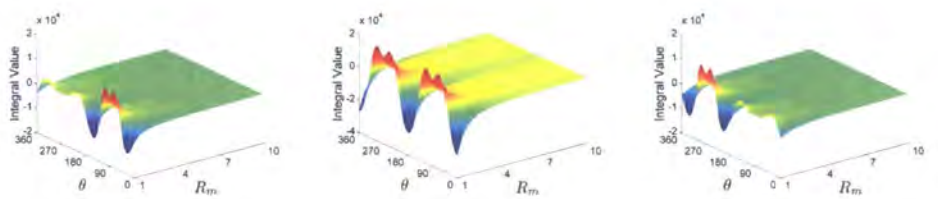


(c) End-node 2

Figure B.4: $h_{\zeta\zeta}$

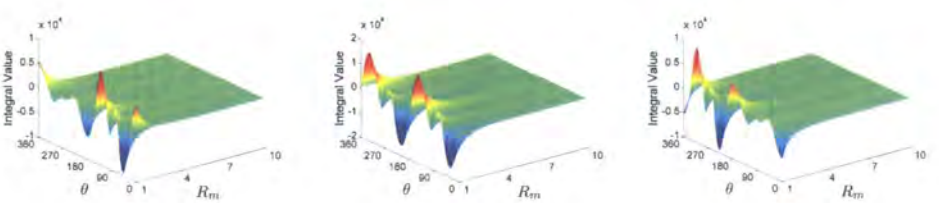
APPENDIX C

Surface Plots - s terms



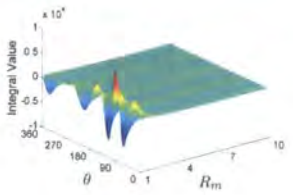
(a) End-node 1 (b) Mid-node (c) End-node 2

Figure C.1: $s_{1\eta\eta}$

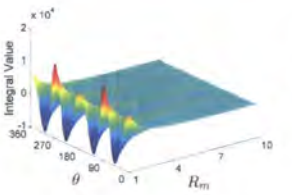


(a) End-node 1 (b) Mid-node (c) End-node 2

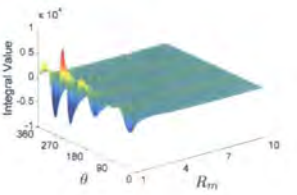
Figure C.2: $s_{1\eta\zeta}$



(a) End-node 1

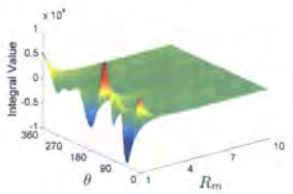


(b) Mid-node

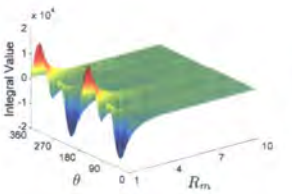


(c) End-node 2

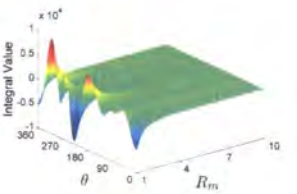
Figure C.3: $s_{1\zeta\zeta}$



(a) End-node 1

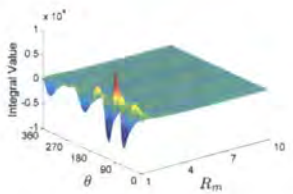


(b) Mid-node

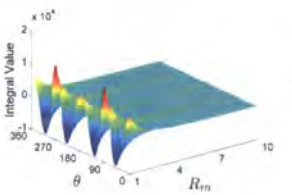


(c) End-node 2

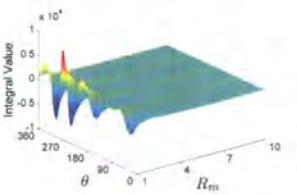
Figure C.4: $s_{2\eta\eta}$



(a) End-node 1

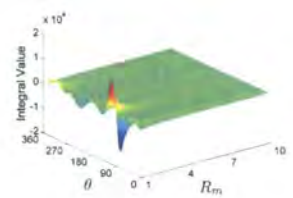


(b) Mid-node

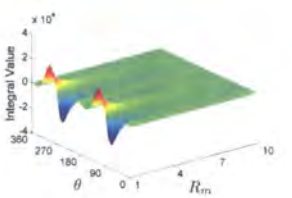


(c) End-node 2

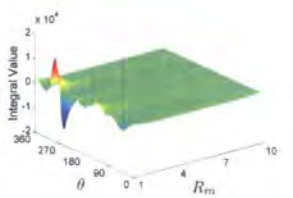
Figure C.5: $s_{2\eta\zeta}$



(a) End-node 1



(b) Mid-node

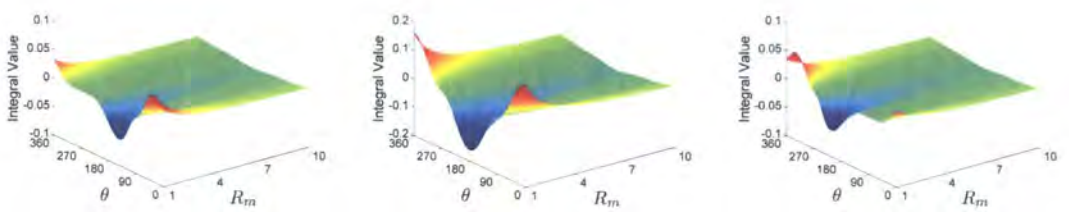


(c) End-node 2

Figure C.6: $s_{2\zeta\zeta}$

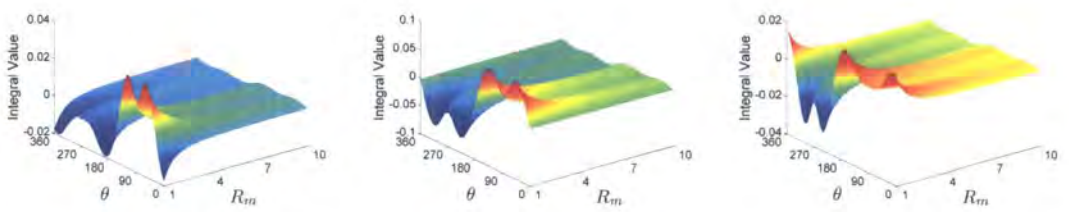
APPENDIX D

Surface Plots - d terms



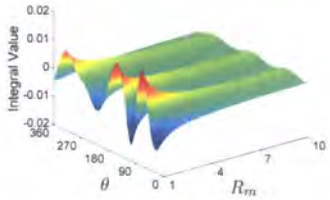
(a) End-node 1 (b) Mid-node (c) End-node 2

Figure D.1: $d_{1\eta\eta}$

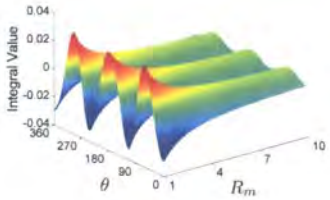


(a) End-node 1 (b) Mid-node (c) End-node 2

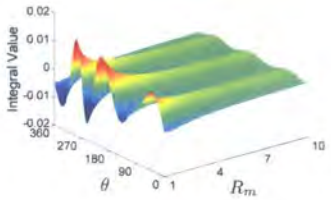
Figure D.2: $d_{1\eta\zeta}$



(a) End-node 1

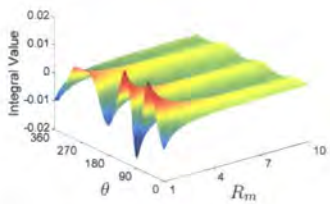


(b) Mid-node

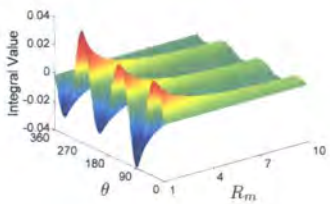


(c) End-node 2

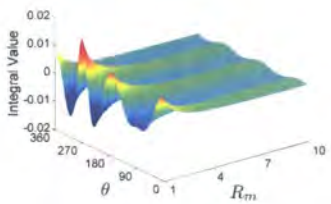
Figure D.3: $d_{1\zeta\zeta}$



(a) End-node 1

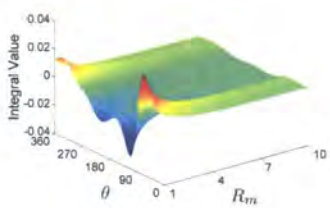


(b) Mid-node

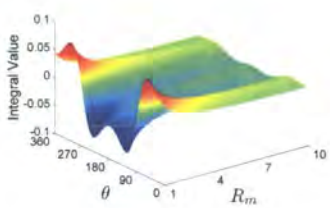


(c) End-node 2

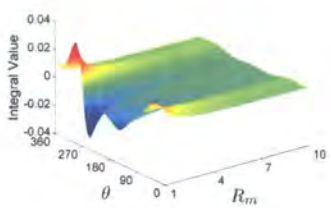
Figure D.4: $d_{2\eta\eta}$



(a) End-node 1

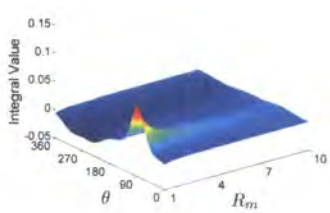


(b) Mid-node

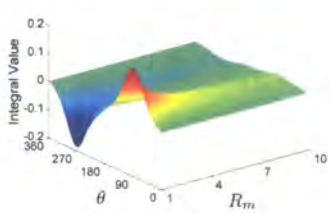


(c) End-node 2

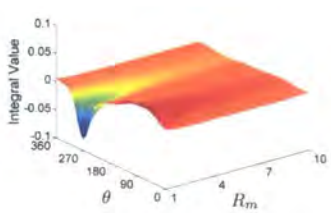
Figure D.5: $d_{2\eta\zeta}$



(a) End-node 1



(b) Mid-node

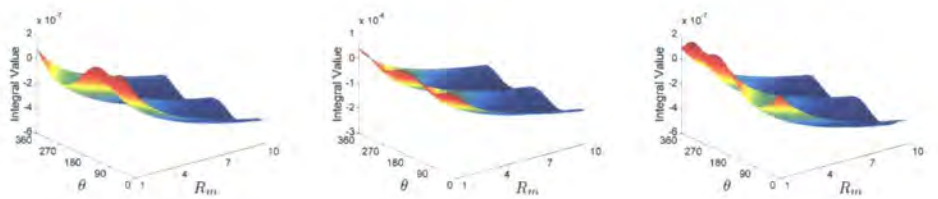


(c) End-node 2

Figure D.6: $d_{2\zeta\zeta}$

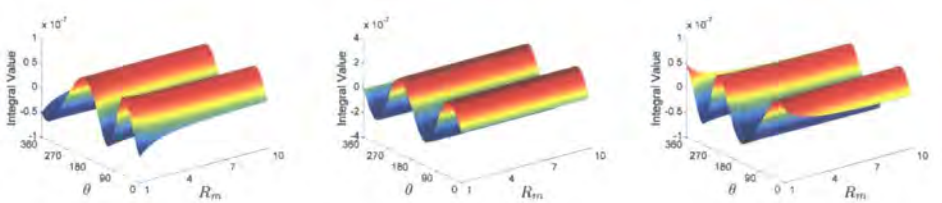
APPENDIX E

Arc Surface Plots - g terms



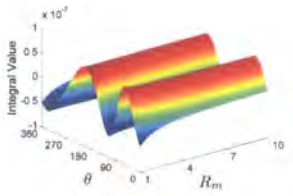
(a) End-node 1 (b) Mid-node (c) End-node 2

Figure E.1: $g_{\eta\eta}$

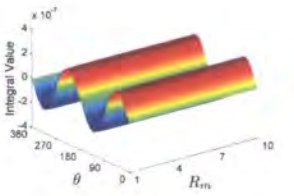


(a) End-node 1 (b) Mid-node (c) End-node 2

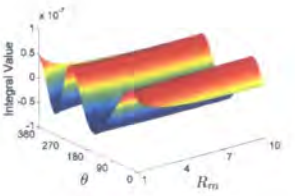
Figure E.2: $g_{\eta\zeta}$ which is identical to $g_{\zeta\eta}$



(a) End-node 1

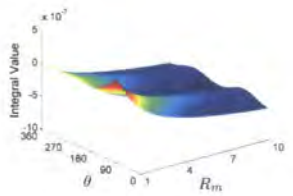


(b) Mid-node

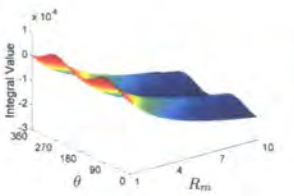


(c) End-node 2

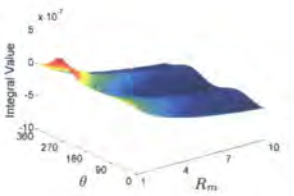
Figure E.3: $g_{\zeta\eta}$



(a) End-node 1



(b) Mid-node

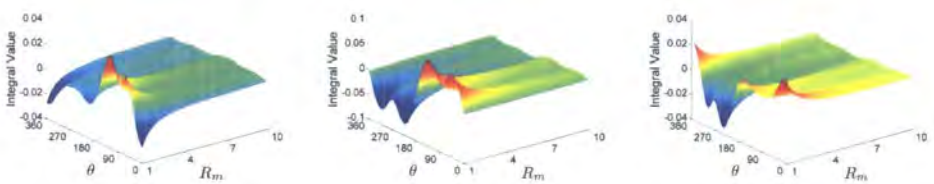


(c) End-node 2

Figure E.4: $g_{\zeta\zeta}$

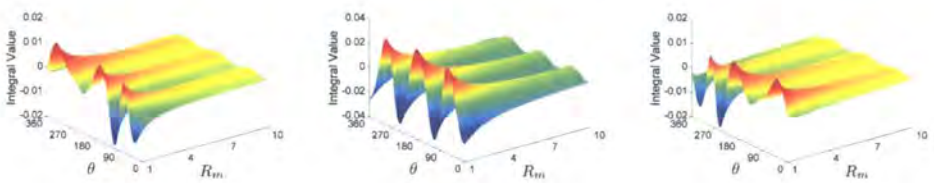
APPENDIX F

Arc Surface Plots - h terms



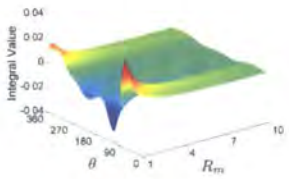
(a) End-node 1 (b) Mid-node (c) End-node 2

Figure F.1: $h_{\eta\eta}$

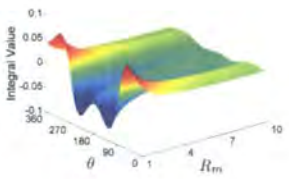


(a) End-node 1 (b) Mid-node (c) End-node 2

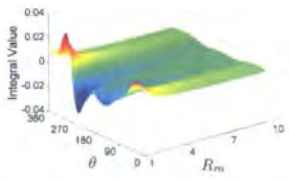
Figure F.2: $h_{\eta\zeta}$



(a) End-node 1

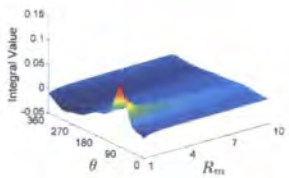


(b) Mid-node

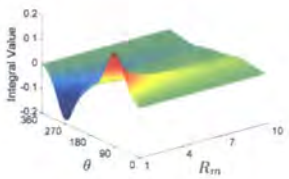


(c) End-node 2

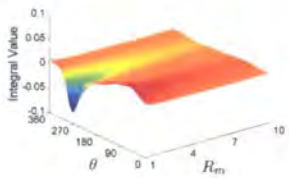
Figure F.3: $h_{\zeta\eta}$



(a) End-node 1



(b) Mid-node

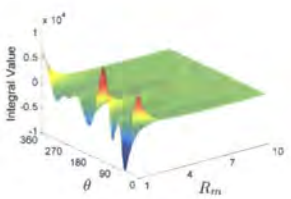


(c) End-node 2

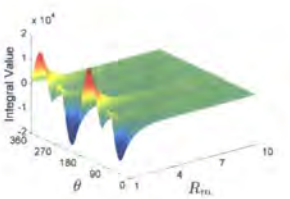
Figure F.4: $h_{\zeta\zeta}$

APPENDIX G

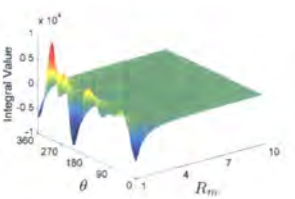
Arc Surface Plots - s terms



(a) End-node 1

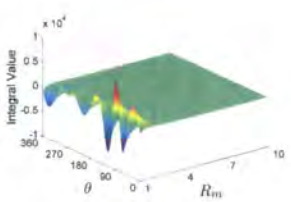


(b) Mid-node

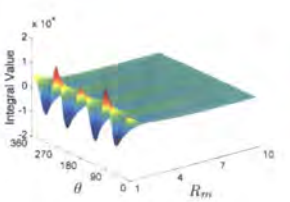


(c) End-node 2

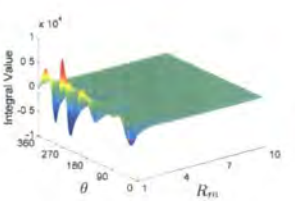
Figure G.1: $s_{1\eta\eta}$



(a) End-node 1

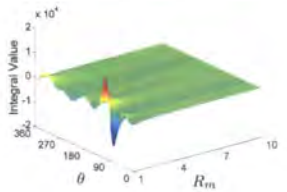


(b) Mid-node

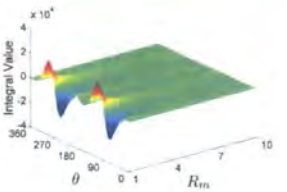


(c) End-node 2

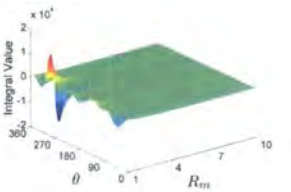
Figure G.2: $s_{1\eta\zeta}$



(a) End-node 1

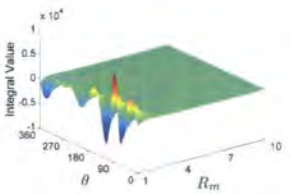


(b) Mid-node

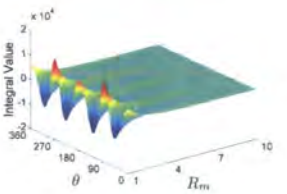


(c) End-node 2

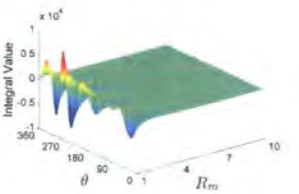
Figure G.3: $s_{1\zeta\zeta}$



(a) End-node 1

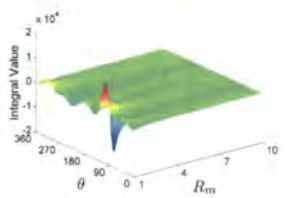


(b) Mid-node

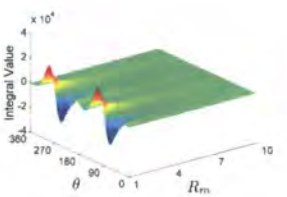


(c) End-node 2

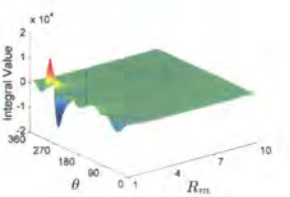
Figure G.4: $s_{2\eta\eta}$



(a) End-node 1

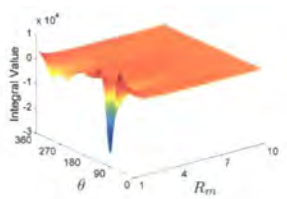


(b) Mid-node

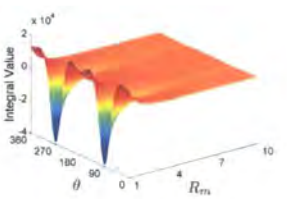


(c) End-node 2

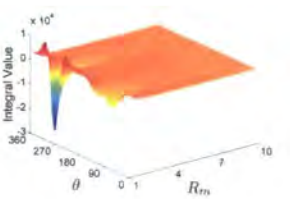
Figure G.5: $s_{2\eta\zeta}$



(a) End-node 1



(b) Mid-node

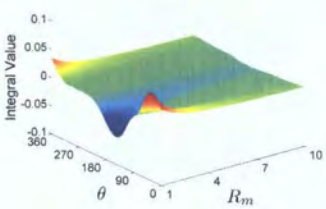


(c) End-node 2

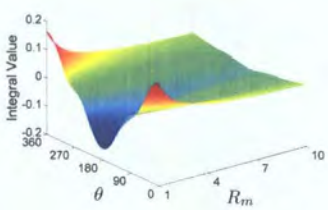
Figure G.6: $s_{2\zeta\zeta}$

APPENDIX H

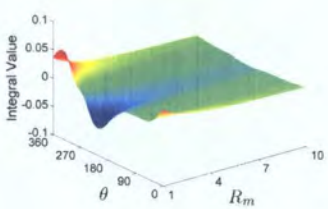
Arc Surface Plots - d terms



(a) End-node 1

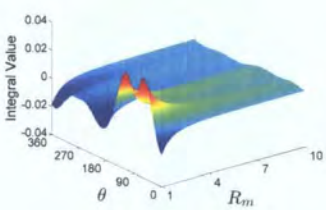


(b) Mid-node

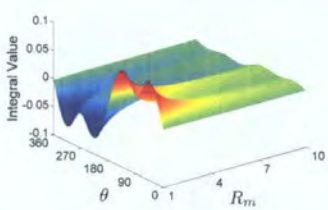


(c) End-node 2

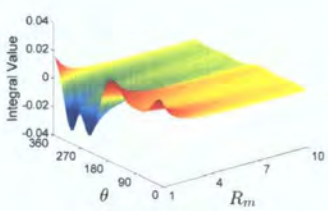
Figure H.1: $d_{1\eta\eta}$



(a) End-node 1

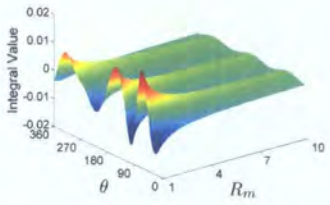


(b) Mid-node

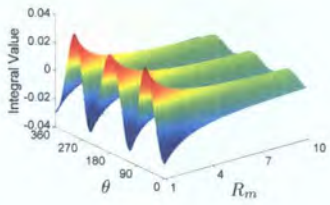


(c) End-node 2

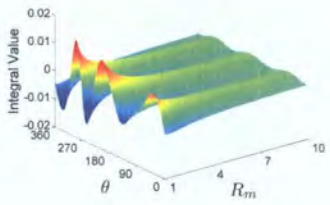
Figure H.2: $d_{1\eta\zeta}$



(a) End-node 1

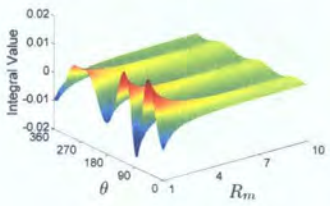


(b) Mid-node

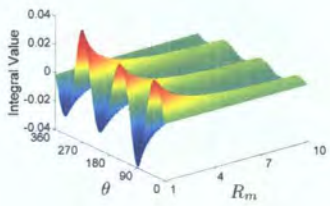


(c) End-node 2

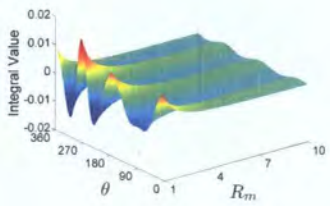
Figure H.3: $d_{1\zeta\zeta}$



(a) End-node 1

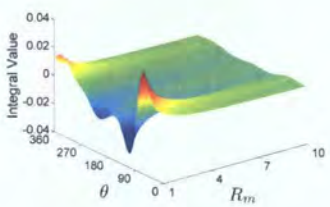


(b) Mid-node

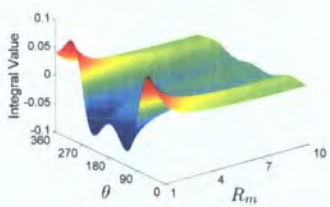


(c) End-node 2

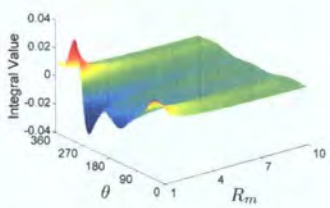
Figure H.4: $d_{2\eta\eta}$



(a) End-node 1

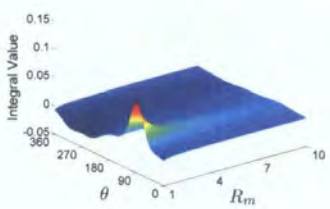


(b) Mid-node

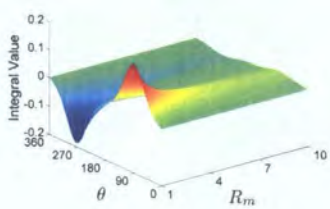


(c) End-node 2

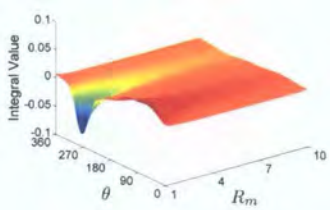
Figure H.5: $d_{2\eta\zeta}$



(a) End-node 1



(b) Mid-node



(c) End-node 2

Figure H.6: $d_{2\zeta\zeta}$

APPENDIX I

Surface Fit Equations - g terms

The notation used in this section is g_{ijk} where i is the source point direction, j is the field element direction and k is the node number on the field element.

I.1 $\phi = 0,\ 2 < R_m < 3$

$$g_{001} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249 \ln R_m \\ +R_m^{-1}\,(0.854\cos(\theta) + 0.273\cos(3\theta)) \\ +R_m^{-2}\,(0.087\cos(2\theta) + 0.081\cos(4\theta)) + 0.040 R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-7}$$

$$g_{002} = [2.166 + 2.185\cos(2\theta) - 8.996 \ln R_m + 0.108 R_m^{-2}\cos(4\theta)] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249 \ln R_m \\ -R_m^{-1}\,(0.854\cos(\theta) + 0.273\cos(3\theta)) \\ +R_m^{-2}\,(0.087\cos(2\theta) + 0.081\cos(4\theta)) - 0.040 R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414\sin(2\theta) - 2.707 R_m^{-1}\,(\sin(\theta) - \sin(3\theta)) \\ -R_m^{-2}\,(0.812\sin(2\theta) - 0.785\sin(4\theta)) \\ -R_m^{-3}\,(0.406\sin(3\theta) - 0.392\sin(5\theta)) + 0.139 R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = \left[\sin(2\theta)\left(2.166 - 0.108R_m^{-2}\right) + 0.106R_m^{-2}\sin(4\theta)\right] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414\sin(2\theta) + 2.707R_m^{-1}\left(\sin(\theta) - \sin(3\theta)\right) \\ -R_m^{-2}\left(0.812\sin(2\theta) - 0.785\sin(4\theta)\right) \\ +R_m^{-3}\left(0.406\sin(3\theta) - 0.392\sin(5\theta)\right) + 0.139R_m^{-4}\sin(6\theta) \end{array}\right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541\left(1 - \cos(2\theta)\right) - 2.249\ln R_m \\ +R_m^{-1}\left(1.395\cos(\theta) - 0.271\cos(3\theta)\right) \\ +R_m^{-2}\left(0.250\cos(2\theta) - 0.076\cos(4\theta)\right) \\ +R_m^{-3}\left(0.097\cos(3\theta) - 0.038\cos(5\theta)\right) \end{array}\right] \times 10^{-7}$$

$$g_{112} = \left[\begin{array}{l} 2.166 - 8.996\ln R_m - 0.104R_m^{-2}\cos(4\theta) \\ -\cos(2\theta)\left(0.903\ln R_m + 4.415R_m^{-1} - 3.009R_m^{-2}\right) \end{array}\right] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541\left(1 - \cos(2\theta)\right) - 2.249\ln R_m \\ -R_m^{-1}\left(1.395\cos(\theta) - 0.271\cos(3\theta)\right) \\ +R_m^{-2}\left(0.250\cos(2\theta) - 0.076\cos(4\theta)\right) \\ -R_m^{-3}\left(0.097\cos(3\theta) - 0.038\cos(5\theta)\right) \end{array}\right] \times 10^{-7}$$

I.2 $\phi = 0,\ 3 < R_m < 15$

$$g_{001} = \left[\begin{array}{l} 0.541 + 0.543\cos(2\theta) - 2.249\ln R_m \\ +R_m^{-1}\left(0.854\cos(\theta) + 0.271\cos(3\theta)\right) \end{array}\right] \times 10^{-7}$$

$$g_{002} = \left[2.166 + 2.168\cos(2\theta) - 8.996\ln R_m\right] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541 + 0.543\cos(2\theta) - 2.249\ln R_m \\ -R_m^{-1}\left(0.854\cos(\theta) + 0.271\cos(3\theta)\right) \end{array}\right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414\sin(2\theta) - R_m^{-1}\left(2.707\sin(\theta) - 2.688\sin(3\theta)\right) \\ -R_m^{-2}\left(0.812\sin(2\theta) - 0.802\sin(4\theta)\right) + 0.400R_m^{-3}\sin(5\theta) \end{array}\right] \times 10^{-8}$$

$$g_{012} = \left[2.166\sin(2\theta) - 0.108R_m^{-2}\left(\sin(2\theta) - \sin(4\theta)\right)\right] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414\sin(2\theta) + R_m^{-1}(2.707\sin(\theta) - 2.688\sin(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) - 0.802\sin(4\theta)) - 0.400R_m^{-3}\sin(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541 - \ln R_m\,(2.249 + 0.345\cos(2\theta)) + 0.060\ln R_m^2\cos(2\theta) \\ +R_m^{-1}(1.395\cos(\theta) - 0.644\cos(2\theta) - 0.266\cos(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{112} = [2.166 - 2.158\cos(2\theta) - 8.996\ln R_m] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541 - \ln R_m\,(2.249 + 0.345\cos(2\theta)) + 0.060\ln R_m^2\cos(2\theta) \\ -R_m^{-2}(1.395\cos(\theta) + 0.644\cos(2\theta) - 0.266\cos(3\theta)) \end{array} \right] \times 10^{-7}$$

I.3 $\phi = 90,\ 2 < R_m < 3$

$$g_{001} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249\ln R_m \\ +R_m^{-1}(1.395\sin(\theta) + 0.271\sin(3\theta)) \\ -R_m^{-2}(0.250\cos(2\theta) + 0.076\cos(4\theta)) \\ -R_m^{-3}(0.097\sin(3\theta) + 0.038\sin(5\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{002} = \left[\begin{array}{l} 2.166\,(1 + \cos(2\theta)) - 8.996\ln R_m \\ -R_m^{-2}(0.333\cos(2\theta) + 0.104\cos(4\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249\ln R_m \\ -R_m^{-1}(1.395\sin(\theta) + 0.271\sin(3\theta)) \\ -R_m^{-2}(0.250\cos(2\theta) + 0.076\cos(4\theta)) \\ +R_m^{-3}(0.097\sin(3\theta) + 0.038\sin(5\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414\sin(2\theta) - 2.707R_m^{-1}(\cos(\theta) + \cos(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) + 0.785\sin(4\theta)) \\ +R_m^{-3}(0.406\cos(3\theta) + 0.392\cos(5\theta)) \\ +0.139R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = [\sin(2\theta)\,(1.825\ln R_m - 0.490\ln R_m^2 + 2.220R_m^{-1}) - 0.106R_m^{-2}\sin(4\theta)] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414\sin(2\theta) + 2.707R_m^{-1}(\cos(\theta) + \cos(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) + 0.785\sin(4\theta)) \\ -R_m^{-3}(0.406\cos(3\theta) + 0.392\cos(5\theta)) \\ +0.139R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541(1 - \cos(2\theta)) - 2.249\ln R_m \\ +R_m^{-1}(0.854\sin(\theta) - 0.273\sin(3\theta)) \\ -R_m^{-2}(0.087\cos(2\theta) - 0.081\cos(4\theta)) + 0.040R_m^{-3}\sin(5\theta) \end{array} \right] \times 10^{-7}$$

$$g_{112} = \left[\begin{array}{l} 2.166 - 8.996\ln R_m + 0.108R_m^{-2}\cos(4\theta) \\ -\cos(2\theta)(1.751\ln R_m - 0.451\ln R_m^2 + 2.397R_m^{-1}) \end{array} \right] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541(1 - \cos(2\theta)) - 2.249\ln R_m \\ -R_m^{-1}(0.854\sin(\theta) - 0.273\sin(3\theta)) \\ -R_m^{-2}(0.087\cos(2\theta) - 0.081\cos(4\theta)) - 0.040R_m^{-3}\sin(5\theta) \end{array} \right] \times 10^{-7}$$

I.4 $\phi = 90,\ 3 < R_m < 15$

$$g_{001} = \left[\begin{array}{l} 0.541(1 + \cos(2\theta)) - 2.249\ln R_m \\ +R_m^{-1}(1.395\sin(\theta) + 0.266\sin(3\theta)) - 0.250R_m^{-2}\cos(2\theta) \end{array} \right] \times 10^{-7}$$

$$g_{002} = [2.166 + 2.158\cos(2\theta) - 8.996\ln R_m] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541(1 + \cos(2\theta)) - 2.249\ln R_m \\ -R_m^{-1}(1.395\sin(\theta) + 0.266\sin(3\theta)) - 0.250R_m^{-2}\cos(2\theta) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414\sin(2\theta) - R_m^{-1}(2.707\cos(\theta) + 2.688\cos(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) + 0.802\sin(4\theta)) + 0.400R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = [2.166\sin(2\theta) - 0.108R_m^{-2}(\sin(2\theta) + \sin(4\theta))] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414\sin(2\theta) + R_m^{-1}(2.707\cos(\theta) + 2.688\cos(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) + 0.802\sin(4\theta)) - 0.400R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541 - 0.543\cos(2\theta) - 2.249\ln R_m \\ +R_m^{-1}(0.854\sin(\theta) - 0.271\sin(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{112} = [2.166(1 - \cos(2\theta)) - 8.996\ln R_m] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541 - 0.543\cos(2\theta) - 2.249\ln R_m \\ -R_m^{-1}(0.854\sin(\theta) - 0.271\sin(3\theta)) \end{array} \right] \times 10^{-7}$$

I.5 $\phi = 180,\ 2 < R_m < 3$

$$g_{001} = \left[\begin{array}{l} 0.541(1 + \cos(2\theta)) - 2.249\ln R_m \\ -R_m^{-1}(0.854\cos(\theta) + 0.273\cos(3\theta)) \\ +R_m^{-2}(0.087\cos(2\theta) + 0.081\cos(4\theta)) - 0.040R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-7}$$

$$g_{002} = [2.166 + 2.185\cos(2\theta) - 8.996\ln R_m + 0.108R_m^{-2}\cos(4\theta)] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541(1 + \cos(2\theta)) - 2.249\ln R_m \\ +R_m^{-1}(0.854\cos(\theta) + 0.273\cos(3\theta)) \\ +R_m^{-2}(0.087\cos(2\theta) + 0.081\cos(4\theta)) + 0.040R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414\sin(2\theta) + 2.707R_m^{-1}(\sin(\theta) - \sin(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) - 0.785\sin(4\theta)) \\ +R_m^{-3}(0.406\sin(3\theta) - 0.392\sin(5\theta)) + 0.139R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = [\sin(2\theta)(2.166 - 0.108R_m^{-2}) + 0.106R_m^{-2}\sin(4\theta)] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414\sin(2\theta) - 2.707R_m^{-1}(\sin(\theta) - \sin(3\theta)) \\ -R_m^{-2}(0.812\sin(2\theta) - 0.785\sin(4\theta)) \\ -R_m^{-3}(0.406\sin(3\theta) - 0.392\sin(5\theta)) + 0.139R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541(1 - \cos(2\theta)) - 2.249\ln R_m \\ -R_m^{-1}(1.395\cos(\theta) - 0.271\cos(3\theta)) \\ +R_m^{-2}(0.250\cos(2\theta) - 0.076\cos(4\theta)) \\ -R_m^{-3}(0.097\cos(3\theta) - 0.038\cos(5\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{112} = \left[\begin{array}{l} 2.166 - 8.996 \ln R_m - 0.104 R_m^{-2} \cos(4\theta) \\ -\cos(2\theta)\,(0.903 \ln R_m + 4.415 R_m^{-1} - 3.009 R_m^{-2}) \end{array} \right] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541\,(1 - \cos(2\theta)) - 2.249 \ln R_m \\ +R_m^{-1}\,(1.395 \cos(\theta) - 0.271 \cos(3\theta)) \\ +R_m^{-2}\,(0.250 \cos(2\theta) - 0.076 \cos(4\theta)) \\ +R_m^{-3}\,(0.097 \cos(3\theta) - 0.038 \cos(5\theta)) \end{array} \right] \times 10^{-7}$$

I.6 $\phi = 180,\ 3 < R_m < 15$

$$g_{001} = \left[\begin{array}{l} 0.541 + 0.543 \cos(2\theta) - 2.249 \ln R_m \\ -R_m^{-1}\,(0.854 \cos(\theta) + 0.271 \cos(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{002} = [2.166 + 2.168 \cos(2\theta) - 8.996 \ln R_m] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541 + 0.543 \cos(2\theta) - 2.249 \ln R_m \\ +R_m^{-1}\,(0.854 \cos(\theta) + 0.271 \cos(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414 \sin(2\theta) + R_m^{-1}\,(2.707 \sin(\theta) - 2.688 \sin(3\theta)) \\ -R_m^{-2}\,(0.812 \sin(2\theta) - 0.802 \sin(4\theta)) \\ -0.400 R_m^{-3} \sin(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = [2.166 \sin(2\theta) - 0.108 R_m^{-2}\,(\sin(2\theta) - \sin(4\theta))] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414 \sin(2\theta) - R_m^{-1}\,(2.707 \sin(\theta) - 2.688 \sin(3\theta)) \\ -R_m^{-2}\,(0.812 \sin(2\theta) - 0.802 \sin(4\theta)) \\ +0.400 R_m^{-3} \sin(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541 - \ln R_m\,(2.249 - 0.345 \cos(2\theta)) + 0.060 \ln R_m^2 \cos(2\theta) \\ -R_m^{-1}\,(1.395 \cos(\theta) + 0.644 \cos(2\theta) - 0.266 \cos(3\theta)) \end{array} \right] \times 10^{-8}$$

$$g_{112} = [2.166 - 2.158 \cos(2\theta) - 8.996 \ln R_m] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541 - \ln R_m \left(2.249 + 0.345 \cos\left(2\theta\right)\right) + 0.060 \ln R_m^2 \cos\left(2\theta\right) \\ +R_m^{-1} \left(1.395 \cos\left(\theta\right) - 0.644 \cos\left(2\theta\right) - 0.266 \cos\left(3\theta\right)\right) \end{array} \right] \times 10^{-8}$$

I.7 $\phi = 270,\ 2 < R_m < 3$

$$g_{001} = \left[\begin{array}{l} 0.541 \left(1 + \cos\left(2\theta\right)\right) - 2.249 \ln R_m \\ -R_m^{-1} \left(1.395 \sin\left(\theta\right) + 0.271 \sin\left(3\theta\right)\right) \\ -R_m^{-2} \left(0.250 \cos\left(2\theta\right) + 0.076 \cos\left(4\theta\right)\right) \\ +R_m^{-3} \left(0.097 \sin\left(3\theta\right) + 0.038 \sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-7}$$

$$g_{002} = \left[\begin{array}{l} 2.166 \left(1 + \cos\left(2\theta\right)\right) - 8.996 \ln R_m \\ -R_m^{-2} \left(0.333 \cos\left(2\theta\right) + 0.104 \cos\left(4\theta\right)\right) \end{array} \right] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541 \left(1 + \cos\left(2\theta\right)\right) - 2.249 \ln R_m \\ +R_m^{-1} \left(1.395 \sin\left(\theta\right) + 0.271 \sin\left(3\theta\right)\right) \\ -R_m^{-2} \left(0.250 \cos\left(2\theta\right) + 0.076 \cos\left(4\theta\right)\right) \\ -R_m^{-3} \left(0.097 \sin\left(3\theta\right) + 0.038 \sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414 \sin\left(2\theta\right) + 2.707 R_m^{-1} \left(\cos\left(\theta\right) + \cos\left(3\theta\right)\right) \\ -R_m^{-2} \left(0.812 \sin\left(2\theta\right) + 0.785 \sin\left(4\theta\right)\right) \\ -R_m^{-3} \left(0.406 \cos\left(3\theta\right) + 0.392 \cos\left(5\theta\right)\right) + 0.139 R_m^{-4} \sin\left(6\theta\right) \end{array} \right] \times 10^{-8}$$

$$g_{012} = \left[\sin\left(2\theta\right) \left(1.825 \ln R_m - 0.490 \ln R_m^2 + 2.220 R_m^{-1}\right) - 0.106 R_m^{-2} \sin\left(4\theta\right)\right] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414 \sin\left(2\theta\right) - 2.707 R_m^{-1} \left(\cos\left(\theta\right) + \cos\left(3\theta\right)\right) \\ -R_m^{-2} \left(0.812 \sin\left(2\theta\right) + 0.785 \sin\left(4\theta\right)\right) \\ +R_m^{-3} \left(0.406 \cos\left(3\theta\right) + 0.392 \cos\left(5\theta\right)\right) + 0.139 R_m^{-4} \sin\left(6\theta\right) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541 \left(1 - \cos\left(2\theta\right)\right) - 2.249 \ln R_m \\ -R_m^{-1} \left(0.854 \sin\left(\theta\right) - 0.273 \sin\left(3\theta\right)\right) \\ -R_m^{-2} \left(0.087 \cos\left(2\theta\right) - 0.081 \cos\left(4\theta\right)\right) - 0.040 R_m^{-3} \sin\left(5\theta\right) \end{array} \right] \times 10^{-7}$$

$$g_{112} = \left[\begin{array}{l} 2.166 - 8.996 \ln R_m + 0.108 R_m^{-2} \cos\left(4\theta\right) \\ -\cos\left(2\theta\right) \left(1.751 \ln R_m - 0.451 \ln R_m^2 + 2.397 R_m^{-1}\right) \end{array} \right] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541\,(1 - \cos(2\theta)) - 2.249 \ln R_m \\ +R_m^{-1}\,(0.854 \sin(\theta) - 0.273 \sin(3\theta)) \\ -R_m^{-2}\,(0.087 \cos(2\theta) - 0.081 \cos(4\theta)) + 0.040 R_m^{-3} \sin(5\theta) \end{array} \right] \times 10^{-7}$$

I.8 $\phi = 270,\ 3 < R_m < 15$

$$g_{001} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249 \ln R_m - 0.250 R_m^{-2} \cos(2\theta) \\ -R_m^{-1}\,(1.395 \sin(\theta) + 0.266 \sin(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{002} = [2.166 + 2.158 \cos(2\theta) - 8.996 \ln R_m] \times 10^{-7}$$

$$g_{003} = \left[\begin{array}{l} 0.541\,(1 + \cos(2\theta)) - 2.249 \ln R_m - 0.250 R_m^{-2} \cos(2\theta) \\ +R_m^{-1}\,(1.395 \sin(\theta) + 0.266 \sin(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{011} = \left[\begin{array}{l} 5.414 \sin(2\theta) + R_m^{-1}\,(2.707 \cos(\theta) + 2.688 \cos(3\theta)) \\ -R_m^{-2}\,(0.812 \sin(2\theta) + 0.802 \sin(4\theta)) - 0.400 R_m^{-3} \cos(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{012} = [2.166 \sin(2\theta) - 0.108 R_m^{-2}\,(\sin(2\theta) + \sin(4\theta))] \times 10^{-7}$$

$$g_{013} = \left[\begin{array}{l} 5.414 \sin(2\theta) - R_m^{-1}\,(2.707 \cos(\theta) + 2.688 \cos(3\theta)) \\ -R_m^{-2}\,(0.812 \sin(2\theta) + 0.802 \sin(4\theta)) + 0.400 R_m^{-3} \cos(5\theta) \end{array} \right] \times 10^{-8}$$

$$g_{111} = \left[\begin{array}{l} 0.541 - 0.543 \cos(2\theta) - 2.249 \ln R_m \\ -R_m^{-1}\,(0.854 \sin(\theta) - 0.271 \sin(3\theta)) \end{array} \right] \times 10^{-7}$$

$$g_{112} = [2.166\,(1 - \cos(2\theta)) - 8.996 \ln R_m] \times 10^{-7}$$

$$g_{113} = \left[\begin{array}{l} 0.541 - 0.543 \cos(2\theta) - 2.249 \ln R_m \\ +R_m^{-1}\,(0.854 \sin(\theta) - 0.271 \sin(3\theta)) \end{array} \right] \times 10^{-7}$$

APPENDIX J

Surface Fit Equations - h terms

The notation used in this section is h_{ijk} where i is the source point direction, j is the field element direction and k is the node number on the field element.

J.1 $\phi = 0,\ 2 < R_m < 3$

$$h_{001} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) + 0.379R_m^{-3}\sin\left(5\theta\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) \\ -0.050\sin\left(7\theta\right) - 0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.451\sin\left(3\theta\right)\right) + 0.510R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{003} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) + 0.379R_m^{-3}\sin\left(5\theta\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) \\ +0.050\sin\left(7\theta\right) - 0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.890\cos\left(4\theta\right) + 0.773\cos\left(5\theta\right) - 2.471\cos\left(6\theta\right) \\ -0.492\cos\left(7\theta\right) - 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{012} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) - 0.040R_m^{-4}\cos\left(7\theta\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.503\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{013} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} -1.890\cos\left(4\theta\right) + 0.773\cos\left(5\theta\right) + 2.471\cos\left(6\theta\right) \\ -0.492\cos\left(7\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{101} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.361\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.328\cos\left(4\theta\right) - 0.242\cos\left(6\theta\right) - 0.048\cos\left(7\theta\right) \\ -0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{102} = \left[\begin{array}{l} R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.569\cos\left(3\theta\right)\right) + 0.587R_m^{-2}\cos\left(3\theta\right) \\ -0.495R_m^{-3}\cos\left(5\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{103} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.361\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} -0.328\cos\left(4\theta\right) + 0.242\cos\left(6\theta\right) - 0.048\cos\left(7\theta\right) \\ +0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{111} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.352\sin\left(5\theta\right) - 0.186\sin\left(6\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.048\sin\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.105\sin\left(3\theta\right) - 0.049\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$h_{113} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.352\sin\left(5\theta\right) + 0.186\sin\left(6\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.048\sin\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

J.2 $\phi = 0,\ 3 < R_m < 15$

$$h_{001} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.449\sin\left(3\theta\right)\right) + 0.514R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{003} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$h_{012} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.511\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{013} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$h_{101} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{102} = \left[\begin{array}{l} R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.449\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.703\cos\left(3\theta\right) - 0.508\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{103} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{111} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.105\sin\left(3\theta\right) - 0.051\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$h_{113} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

J.3 $\phi = 90,\ 2 < R_m < 3$

$$h_{001} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(0.186\sin\left(6\theta\right) + 0.786\cos\left(3\theta\right) + 0.352\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.048\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[\begin{array}{l} -R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.105\cos\left(3\theta\right) + 0.049\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$h_{003} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ -R_m^{-3}\left(0.186\sin\left(6\theta\right) - 0.786\cos\left(3\theta\right) - 0.352\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.048\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.048\sin\left(7\theta\right) + 0.328\cos\left(4\theta\right) \\ +0.242\cos\left(6\theta\right) - 0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{012} = \left[\begin{array}{l} -R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.495\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$h_{013} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.048\sin\left(7\theta\right) - 0.328\cos\left(4\theta\right) \\ -0.242\cos\left(6\theta\right) + 0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{101} = \left[\begin{array}{l} R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\sin\left(3\theta\right) + 4.042\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) \\ +2.471\cos\left(6\theta\right) - 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{102} = \left[\begin{array}{l} R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.503\sin\left(5\theta\right)\right) - 0.040R_m^{-4}\sin\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{103} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 4.042\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) \\ -2.471\cos\left(6\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{111} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\cos(\theta) - 0.864\cos(3\theta)\right) \\ -R_m^{-2}\left(0.464\sin(2\theta) - 0.862\sin(4\theta)\right) - 0.379R_m^{-3}\cos(5\theta) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin(4\theta) + 0.252\sin(6\theta) - 0.014\sin(8\theta) \\ -0.050\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{112} = \left[-R_m^{-1}\left(7.162\cos(\theta) - 3.452\cos(3\theta)\right) - 0.510R_m^{-3}\cos(5\theta)\right] \times 10^{-2}$$

$$h_{113} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\cos(\theta) - 0.864\cos(3\theta)\right) \\ -R_m^{-2}\left(0.464\sin(2\theta) - 0.862\sin(4\theta)\right) - 0.379R_m^{-3}\cos(5\theta) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin(4\theta) + 0.252\sin(6\theta) - 0.014\sin(8\theta) \\ -0.050\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

J.4 $\phi = 90,\ 3 < R_m < 15$

$$h_{001} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\cos(\theta) + 0.862\cos(3\theta)\right) \\ -R_m^{-2}\left(2.188\sin(2\theta) + 0.862\sin(4\theta)\right) \\ +R_m^{-3}\left(0.786\cos(3\theta) + 0.373\cos(5\theta)\right) \\ +R_m^{-4}\left(0.458\sin(4\theta) + 0.250\sin(6\theta)\right) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[\begin{array}{l} -R_m^{-1}\left(1.406\cos(\theta) + 0.345\cos(3\theta)\right) \\ +R_m^{-3}\left(0.105\cos(3\theta) + 0.051\cos(5\theta)\right) \end{array} \right] \times 10^{-1}$$

$$h_{003} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\cos(\theta) + 0.862\cos(3\theta)\right) \\ +R_m^{-2}\left(2.188\sin(2\theta) + 0.862\sin(4\theta)\right) \\ +R_m^{-3}\left(0.786\cos(3\theta) + 0.373\cos(5\theta)\right) \\ -R_m^{-4}\left(0.458\sin(4\theta) + 0.250\sin(6\theta)\right) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin(\theta) + 0.862\sin(3\theta)\right) \\ +R_m^{-2}\left(1.326\cos(2\theta) + 0.862\cos(4\theta)\right) \\ +R_m^{-3}\left(0.527\sin(3\theta) + 0.377\sin(5\theta)\right) \\ -R_m^{-4}\left(0.328\cos(4\theta) + 0.252\cos(6\theta)\right) \end{array} \right] \times 10^{-2}$$

$$h_{012} = \left[\begin{array}{l} -R_m^{-1}\left(7.162\sin(\theta) + 3.448\sin(3\theta)\right) \\ +R_m^{-3}\left(0.703\sin(3\theta) + 0.508\sin(5\theta)\right) \end{array} \right] \times 10^{-2}$$

$$h_{013} = \begin{bmatrix} -R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{101} = \begin{bmatrix} R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.330\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) + 2.537\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$h_{102} = \begin{bmatrix} R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.512\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{103} = \begin{bmatrix} R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.330\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$h_{111} = \begin{bmatrix} -R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ -0.384R_m^{-3}\cos\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$h_{112} = \left[-R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.449\cos\left(3\theta\right)\right) - 0.514R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{113} = \begin{bmatrix} -R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ -0.384R_m^{-3}\cos\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

J.5 $\phi = 180,\ 2 < R_m < 3$

$$h_{001} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) - 0.379R_m^{-3}\sin\left(5\theta\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) + 0.050\sin\left(7\theta\right) \\ -0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[-R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.451\sin\left(3\theta\right)\right) - 0.510R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{003} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) - 0.379R_m^{-3}\sin\left(5\theta\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) - 0.050\sin\left(7\theta\right) \\ -0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.890\cos\left(4\theta\right) - 0.773\cos\left(5\theta\right) - 2.471\cos\left(6\theta\right) \\ +0.492\cos\left(7\theta\right) - 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{012} = \left[\begin{array}{l} R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.503\cos\left(5\theta\right)\right) + 0.040R_m^{-4}\cos\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{013} = \left[\begin{array}{l} R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 1.890\cos\left(4\theta\right) + 0.773\cos\left(5\theta\right) + 2.471\cos\left(6\theta\right) \\ -0.492\cos\left(7\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{101} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.361\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.328\cos\left(4\theta\right) - 0.242\cos\left(6\theta\right) + 0.048\cos\left(7\theta\right) \\ -0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{102} = \left[\begin{array}{l} -R_m^{-1}\left(7.162\cos(\theta) - 3.569\cos(3\theta)\right) - 0.587R_m^{-2}\cos(3\theta) \\ +0.495R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-2}$$

$$h_{103} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\cos(\theta) - 0.862\cos(3\theta)\right) \\ -R_m^{-2}\left(1.326\cos(2\theta) - 0.862\cos(4\theta)\right) \\ -R_m^{-3}\left(0.527\cos(3\theta) - 0.361\cos(5\theta)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.328\cos(4\theta) - 0.242\cos(6\theta) - 0.048\cos(7\theta) \\ -0.013\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{111} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\sin(\theta) - 0.862\sin(3\theta)\right) \\ +R_m^{-2}\left(2.188\sin(2\theta) - 0.862\sin(4\theta) + 0.035\sin(6\theta)\right) \\ -R_m^{-3}\left(0.786\sin(3\theta) - 0.352\sin(5\theta) + 0.186\sin(6\theta)\right) \\ +R_m^{-4}\left(0.458\sin(4\theta) + 0.048\sin(7\theta)\right) \end{array} \right] \times 10^{-2}$$

$$h_{112} = \left[\begin{array}{l} -R_m^{-1}\left(1.406\sin(\theta) - 0.345\sin(3\theta)\right) \\ -R_m^{-3}\left(0.105\sin(3\theta) - 0.049\sin(5\theta)\right) \end{array} \right] \times 10^{-1}$$

$$h_{113} = \left[\begin{array}{l} -R_m^{-1}\left(3.515\sin(\theta) - 0.862\sin(3\theta)\right) \\ -R_m^{-2}\left(2.188\sin(2\theta) - 0.862\sin(4\theta) + 0.035\sin(6\theta)\right) \\ -R_m^{-3}\left(0.786\sin(3\theta) - 0.352\sin(5\theta) - 0.186\sin(6\theta)\right) \\ -R_m^{-4}\left(0.458\sin(4\theta) - 0.048\sin(7\theta)\right) \end{array} \right] \times 10^{-2}$$

J.6 $\phi = 180,\ 3 < R_m < 15$

$$h_{001} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin(\theta) + 0.863\sin(3\theta)\right) \\ +R_m^{-2}\left(0.464\sin(2\theta) + 0.858\sin(4\theta)\right) \\ -0.384R_m^{-3}\sin(5\theta) + 0.256R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-2}$$

$$h_{002} = \left[-R_m^{-1}\left(7.162\sin(\theta) + 3.449\sin(3\theta)\right) - 0.514R_m^{-3}\sin(5\theta)\right] \times 10^{-2}$$

$$h_{003} = \left[\begin{array}{l} -R_m^{-1}\left(1.790\sin(\theta) + 0.863\sin(3\theta)\right) \\ -R_m^{-2}\left(0.464\sin(2\theta) + 0.858\sin(4\theta)\right) \\ -0.384R_m^{-3}\sin(5\theta) - 0.256R_m^{-4}\sin(6\theta) \end{array} \right] \times 10^{-2}$$

$$h_{011} = \begin{bmatrix} R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$h_{012} = \begin{bmatrix} R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.511\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{013} = \begin{bmatrix} R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$h_{101} = \begin{bmatrix} -R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{102} = \begin{bmatrix} -R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.448\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\cos\left(3\theta\right) - 0.508\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{103} = \begin{bmatrix} -R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{111} = \begin{bmatrix} -R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{112} = \begin{bmatrix} -R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.105\sin\left(3\theta\right) - 0.051\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-1}$$

$$h_{113} = \begin{bmatrix} -R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

J.7 $\phi = 270,\ 2 < R_m < 3$

$$h_{001} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ -R_m^{-3}\left(0.186\sin\left(6\theta\right) + 0.786\cos\left(3\theta\right) + 0.352\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.048\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{002} = \begin{bmatrix} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.105\cos\left(3\theta\right) + 0.049\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-1}$$

$$h_{003} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right) + 0.035\sin\left(6\theta\right)\right) \\ -R_m^{-3}\left(0.186\sin\left(6\theta\right) + 0.786\cos\left(3\theta\right) + 0.352\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.048\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{011} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right)\right) \\ +R_m^{-4}\begin{pmatrix} 0.048\sin\left(7\theta\right) - 0.328\cos\left(4\theta\right) - 0.242\cos\left(6\theta\right) \\ +0.013\cos\left(8\theta\right) \end{pmatrix} \end{bmatrix} \times 10^{-2}$$

$$h_{012} = \begin{bmatrix} R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.495\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$h_{013} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right)\right) \\ +R_m^{-4}\begin{pmatrix} 0.048\sin\left(7\theta\right) + 0.328\cos\left(4\theta\right) + 0.242\cos\left(6\theta\right) \\ -0.013\cos\left(8\theta\right) \end{pmatrix} \end{bmatrix} \times 10^{-2}$$

$$h_{101} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 4.042\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) \\ -2.471\cos\left(6\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{102} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.503\sin\left(5\theta\right)\right) + 0.040R_m^{-4}\sin\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$h_{103} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 4.042\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) \\ +2.471\cos\left(6\theta\right) - 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$h_{111} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.864\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) + 0.379R_m^{-3}\cos\left(5\theta\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) + 0.252\sin\left(6\theta\right) - 0.014\sin\left(8\theta\right) \\ +0.050\cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$h_{112} = \left[R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.451\cos\left(3\theta\right)\right) + 0.510R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{113} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.864\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) + 0.379R_m^{-3}\cos\left(5\theta\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) + 0.252\sin\left(6\theta\right) - 0.014\sin\left(8\theta\right) \\ -0.050\cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

J.8 $\phi = 270,\ 3 < R_m < 15$

$$h_{001} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{002} = \left[\begin{array}{l} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.105\cos\left(3\theta\right) + 0.051\cos\left(5\theta\right)\right) \end{array}\right] \times 10^{-1}$$

$$h_{003} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.508\sin\left(5\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{101} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-3}$$

$$h_{102} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.512\sin\left(5\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$h_{103} = \begin{bmatrix} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) + 2.537\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$h_{111} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$h_{112} = \left[R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.449\cos\left(3\theta\right)\right) + 0.514R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$h_{113} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

APPENDIX K

Surface Fit Equations - s terms

The notation used in this section is s_{lijk} where l, i and j are the tensor terms and k is the node number on the field element.

K.1 $\phi = 0,\ 2 < R_m < 3$

$$s_{1001} = \left[\begin{array}{l} R_m^{-2} \left(\begin{array}{l} 2.745\left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) + 0.061\sin\left(5\theta\right) \\ +0.124\sin\left(7\theta\right) - 0.015\sin\left(9\theta\right) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 1.373\sin\left(3\theta\right) + 3.896\sin\left(5\theta\right) + 0.057\sin\left(6\theta\right) \\ -0.927\sin\left(7\theta\right) - 0.159\sin\left(8\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.269\sin\left(6\theta\right) + 2.347\sin\left(7\theta\right) + 0.572\sin\left(8\theta\right) \\ +0.145\sin\left(9\theta\right) + 0.016\sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[1.098R_m^{-2}\left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) + R_m^{-4}\left(0.324\sin\left(6\theta\right) + 0.017\sin\left(8\theta\right)\right)\right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} R_m^{-2} \left(\begin{array}{l} 2.745\left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) - 0.061\sin\left(5\theta\right) \\ -0.124\sin\left(7\theta\right) + 0.015\sin\left(9\theta\right) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373\sin\left(3\theta\right) + 3.896\sin\left(5\theta\right) - 0.057\sin\left(6\theta\right) \\ -0.927\sin\left(7\theta\right) + 0.159\sin\left(8\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.269\sin\left(6\theta\right) - 2.347\sin\left(7\theta\right) + 0.572\sin\left(8\theta\right) \\ -0.145\sin\left(9\theta\right) + 0.016\sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} -0.016\cos\left(6\theta\right) - 0.012R_m^{-1}\cos\left(8\theta\right) \\ -R_m^{-2} \left(\begin{array}{l} 2.745\cos\left(4\theta\right) - 0.090\cos\left(5\theta\right) - 0.652\cos\left(6\theta\right) \\ +0.106\cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 1.373\cos\left(3\theta\right) - 4.768\cos\left(5\theta\right) - 2.316\cos\left(6\theta\right) \\ +0.823\cos\left(7\theta\right) + 0.352\cos\left(8\theta\right) + 0.071\cos\left(9\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 1.235\cos\left(4\theta\right) + 1.558\cos\left(5\theta\right) - 2.180\cos\left(7\theta\right) \\ -0.867\cos\left(8\theta\right) - 0.227\cos\left(9\theta\right) - 0.016\cos\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = \left[\begin{array}{l} -1.098R_m^{-2}\cos\left(4\theta\right) + 0.013R_m^{-3}\cos\left(8\theta\right) \\ +R_m^{-4}\left(0.165\cos\left(4\theta\right) - 0.318\cos\left(6\theta\right) - 0.046\cos\left(8\theta\right)\right) \end{array} \right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} -0.016\cos\left(6\theta\right) - 0.012R_m^{-1}\cos\left(8\theta\right) \\ -R_m^{-2} \left(\begin{array}{l} 2.745\cos\left(4\theta\right) + 0.090\cos\left(5\theta\right) - 0.652\cos\left(6\theta\right) \\ -0.106\cos\left(7\theta\right) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373\cos\left(3\theta\right) - 4.768\cos\left(5\theta\right) + 2.316\cos\left(6\theta\right) \\ +0.823\cos\left(7\theta\right) - 0.352\cos\left(8\theta\right) + 0.071\cos\left(9\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 1.235\cos\left(4\theta\right) - 1.558\cos\left(5\theta\right) + 2.180\cos\left(7\theta\right) \\ -0.867\cos\left(8\theta\right) + 0.227\cos\left(9\theta\right) - 0.016\cos\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.093\sin\left(6\theta\right) + 0.143\sin\left(7\theta\right)\right) \\ +R_m^{-2} \left(\begin{array}{l} 2.745\left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) + 0.149\sin\left(5\theta\right) \\ +0.776\sin\left(6\theta\right) + 0.955\sin\left(7\theta\right) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 4.118\sin\left(3\theta\right) - 5.202\sin\left(5\theta\right) - 2.278\sin\left(6\theta\right) \\ -1.802\sin\left(7\theta\right) + 0.147\sin\left(8\theta\right) + 0.069\sin\left(9\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.471\sin\left(4\theta\right) + 2.597\sin\left(5\theta\right) - 0.538\sin\left(8\theta\right) \\ -0.222\sin\left(9\theta\right) - 0.016\sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1112} = \left[\begin{array}{l} 1.098 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) \\ + R_m^{-4} \left(0.329 \sin\left(4\theta\right) - 0.312 \sin\left(6\theta\right) - 0.017 \sin\left(8\theta\right)\right) \end{array} \right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} - R_m^{-1} \left(0.093 \sin\left(6\theta\right) - 0.143 \sin\left(7\theta\right)\right) \\ + R_m^{-2} \left(\begin{array}{l} 2.745 \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) - 0.149 \sin\left(5\theta\right) \\ + 0.776 \sin\left(6\theta\right) - 0.955 \sin\left(7\theta\right) \end{array} \right) \\ - R_m^{-3} \left(\begin{array}{l} 4.118 \sin\left(3\theta\right) - 5.202 \sin\left(5\theta\right) + 2.278 \sin\left(6\theta\right) \\ - 1.802 \sin\left(7\theta\right) - 0.147 \sin\left(8\theta\right) + 0.069 \sin\left(9\theta\right) \end{array} \right) \\ + R_m^{-4} \left(\begin{array}{l} 2.471 \sin\left(4\theta\right) - 2.597 \sin\left(5\theta\right) - 0.538 \sin\left(8\theta\right) \\ + 0.222 \sin\left(9\theta\right) - 0.016 \sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} 0.140 \cos\left(6\theta\right) - 0.962 R_m^{-1} \cos\left(6\theta\right) \\ - R_m^{-2} \left(\begin{array}{l} 5.491 \cos\left(2\theta\right) - 2.745 \cos\left(4\theta\right) - 1.889 \cos\left(6\theta\right) \\ + 0.014 \cos\left(9\theta\right) \end{array} \right) \\ - R_m^{-3} \left(\begin{array}{l} 6.864 \cos\left(3\theta\right) - 4.637 \cos\left(5\theta\right) + 0.271 \cos\left(7\theta\right) \\ + 0.141 \cos\left(8\theta\right) \end{array} \right) \\ - R_m^{-4} \left(\begin{array}{l} 3.706 \cos\left(4\theta\right) + 2.460 \cos\left(5\theta\right) - 1.442 \cos\left(7\theta\right) \\ - 0.520 \cos\left(8\theta\right) - 0.136 \cos\left(9\theta\right) - 0.015 \cos\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[\begin{array}{l} -0.010 R_m^{-1} \cos(6\theta) - R_m^{-2} \left(2.196 \cos(2\theta) - 1.098 \cos(4\theta)\right) \\ +R_m^{-3} \left(0.186 \cos(6\theta) + 0.007 \cos(8\theta)\right) - 0.494 R_m^{-4} \cos(4\theta) \end{array} \right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} 0.140 \cos(6\theta) - 0.962 R_m^{-1} \cos(6\theta) \\ -R_m^{-2} \left(\begin{array}{l} 5.491 \cos(2\theta) - 2.745 \cos(4\theta) - 1.889 \cos(6\theta) \\ -0.014 \cos(9\theta) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 6.864 \cos(3\theta) - 4.637 \cos(5\theta) + 0.271 \cos(7\theta) \\ -0.141 \cos(8\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 3.706 \cos(4\theta) - 2.460 \cos(5\theta) + 1.442 \cos(7\theta) \\ -0.520 \cos(8\theta) + 0.136 \cos(9\theta) - 0.015 \cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

K.2 $\phi = 0,\ 3 < R_m < 15$

$$s_{1001} = \left[\begin{array}{l} 2.745 R_m^{-2} \left(\sin(2\theta) + \sin(4\theta)\right) \\ +R_m^{-3} \left(\begin{array}{l} 1.373 \sin(3\theta) + 4.143 \sin(5\theta) - 0.120 \sin(7\theta) \\ -0.034 \sin(8\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 0.207 \sin(5\theta) - 2.439 \sin(6\theta) - 1.010 \sin(7\theta) \\ -0.211 \sin(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[1.098 R_m^{-2} \left(\sin(2\theta) + \sin(4\theta)\right) + 0.327 R_m^{-4} \sin(6\theta)\right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} 2.745 R_m^{-2} \left(\sin(2\theta) + \sin(4\theta)\right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373 \sin(3\theta) + 4.143 \sin(5\theta) - 0.120 \sin(7\theta) \\ +0.034 \sin(8\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 0.207 \sin(5\theta) + 2.439 \sin(6\theta) - 1.010 \sin(7\theta) \\ +0.211 \sin(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} -R_m^{-2} \left(2.745 \cos(4\theta) + 0.016 \cos(7\theta)\right) \\ +R_m^{-3} \left(\begin{array}{l} 1.373 \cos(3\theta) - 4.192 \cos(5\theta) - 0.023 \cos(6\theta) \\ +0.255 \cos(7\theta) + 0.033 \cos(8\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 1.235 \cos(4\theta) + 0.620 \cos(5\theta) - 2.327 \cos(6\theta) \\ -1.282 \cos(7\theta) - 0.209 \cos(8\theta) - 0.020 \cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = \left[-1.098R_m^{-2}\cos(4\theta) + R_m^{-4}\left(0.165\cos(4\theta) - 0.324\cos(6\theta)\right)\right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} -R_m^{-2}\left(2.745\cos(4\theta) - 0.016\cos(7\theta)\right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\cos(3\theta) - 4.192\cos(5\theta) + 0.023\cos(6\theta) \\ +0.255\cos(7\theta) - 0.033\cos(8\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.235\cos(4\theta) - 0.620\cos(5\theta) - 2.327\cos(6\theta) \\ +1.282\cos(7\theta) - 0.209\cos(8\theta) + 0.020\cos(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} R_m^{-2}\left(2.745\left(\sin(2\theta) - \sin(4\theta)\right) + 0.017\sin(5\theta)\right) \\ +R_m^{-3}\left(\begin{array}{l} 4.118\sin(3\theta) - 4.393\sin(5\theta) - 0.034\sin(6\theta) \\ +0.114\sin(7\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) + 1.347\sin(5\theta) - 2.255\sin(6\theta) \\ -0.980\sin(7\theta) - 0.092\sin(8\theta) - 0.019\sin(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1112} = \left[1.098R_m^{-2}\left(\sin(2\theta) - \sin(4\theta)\right) + R_m^{-4}\left(0.329\sin(4\theta) - 0.322\sin(6\theta)\right)\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} R_m^{-2}\left(2.745\left(\sin(2\theta) - \sin(4\theta)\right) - 0.017\sin(5\theta)\right) \\ -R_m^{-3}\left(\begin{array}{l} 4.118\sin(3\theta) - 4.393\sin(5\theta) + 0.034\sin(6\theta) \\ +0.114\sin(7\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 1.347\sin(5\theta) - 2.255\sin(6\theta) \\ +0.980\sin(7\theta) - 0.092\sin(8\theta) + 0.019\sin(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$s_{2013} = s_{1113}$

$$s_{2111} = \left[\begin{array}{l} -R_m^{-2}\left(5.491\cos\left(2\theta\right) - 2.745\cos\left(4\theta\right) + 0.024\cos\left(5\theta\right)\right) \\ -R_m^{-3}\left(6.864\cos\left(3\theta\right) - 4.502\cos\left(5\theta\right) + 0.111\cos\left(7\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 3.706\cos\left(4\theta\right) + 1.886\cos\left(5\theta\right) - 2.343\cos\left(6\theta\right) \\ -0.964\cos\left(7\theta\right) - 0.092\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[\begin{array}{l} -R_m^{-2}\left(2.196\cos\left(2\theta\right) - 1.098\cos\left(4\theta\right)\right) \\ -R_m^{-4}\left(0.494\cos\left(4\theta\right) - 0.319\cos\left(6\theta\right)\right) \end{array} \right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} -R_m^{-2}\left(5.491\cos\left(2\theta\right) - 2.745\cos\left(4\theta\right) - 0.024\cos\left(5\theta\right)\right) \\ +R_m^{-3}\left(6.864\cos\left(3\theta\right) - 4.502\cos\left(5\theta\right) + 0.111\cos\left(7\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 3.706\cos\left(4\theta\right) - 1.886\cos\left(5\theta\right) - 2.343\cos\left(6\theta\right) \\ +0.964\cos\left(7\theta\right) - 0.092\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

K.3 $\phi = 90,\ 2 < R_m < 3$

$$s_{1001} = \left[\begin{array}{l} \cos\left(6\theta\right)\left(0.034 - 0.185R_m^{-1}\right) - R_m^{-2}\left(5.491\cos\left(2\theta\right) + 2.745\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 6.864\sin\left(3\theta\right) + 4.637\sin\left(5\theta\right) + 0.271\sin\left(7\theta\right) \\ -0.067\sin\left(9\theta\right) - 1.519\cos\left(6\theta\right) - 0.141\cos\left(8\theta\right) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 2.460\sin\left(5\theta\right) + 1.442\sin\left(7\theta\right) - 0.217\sin\left(9\theta\right) \\ +3.706\cos\left(4\theta\right) - 0.520\cos\left(8\theta\right) + 0.015\cos\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[\begin{array}{l} -0.010R_m^{-1}\cos\left(6\theta\right) + 0.186R_m^{-3}\cos\left(6\theta\right) + 0.494R_m^{-4}\cos\left(4\theta\right) \\ -R_m^{-2}\left(2.196\cos\left(2\theta\right) + 1.098\cos\left(4\theta\right) + 0.003\cos\left(8\theta\right)\right) \end{array} \right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} \cos\left(6\theta\right)\left(0.034 - 0.185R_m^{-1}\right) - R_m^{-2}\left(5.491\cos\left(2\theta\right) + 2.745\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 6.864\sin\left(3\theta\right) + 4.637\sin\left(5\theta\right) + 0.271\sin\left(7\theta\right) \\ -0.067\sin\left(9\theta\right) + 1.519\cos\left(6\theta\right) + 0.141\cos\left(8\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 2.460\sin\left(5\theta\right) + 1.442\sin\left(7\theta\right) - 0.217\sin\left(9\theta\right) \\ -3.706\cos\left(4\theta\right) + 0.520\cos\left(8\theta\right) - 0.015\cos\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} 0.043\sin(6\theta) + 0.057\ln R_m\cos(5\theta) \\ -R_m^{-1}\left(0.227\sin(6\theta) + 0.908\cos(5\theta)\right) \\ -R_m^{-2}\left(\begin{array}{l} 2.745\left(\sin(2\theta)+\sin(4\theta)\right) - 3.460\cos(5\theta) \\ +0.014\cos(9\theta) \end{array}\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.655\sin(6\theta) + 0.147\sin(8\theta) + 4.118\cos(3\theta) \\ +0.293\cos(7\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 0.538\sin(8\theta) + 0.016\sin(10\theta) \\ -1.512\cos(7\theta) + 0.139\cos(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1012} = \left[\begin{array}{l} -R_m^{-2}\left(1.098\left(\sin(2\theta)+\sin(4\theta)\right) + 0.049\sin(6\theta)\right) \\ +0.251R_m^{-3}\sin(6\theta) + R_m^{-4}\left(0.329\sin(4\theta) - 0.017\sin(8\theta)\right) \end{array}\right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} 0.043\sin(6\theta) - 0.057\ln R_m\cos(5\theta) \\ -R_m^{-1}\left(0.227\sin(6\theta) - 0.908\cos(5\theta)\right) \\ -R_m^{-2}\left(\begin{array}{l} 2.745\left(\sin(2\theta)+\sin(4\theta)\right) + 3.460\cos(5\theta) \\ -0.014\cos(9\theta) \end{array}\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.655\sin(6\theta) + 0.147\sin(8\theta) - 4.118\cos(3\theta) \\ -0.293\cos(7\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 0.538\sin(8\theta) + 0.016\sin(10\theta) \\ +1.512\cos(7\theta) - 0.139\cos(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -0.016\cos(6\theta) + 0.005\cos(8\theta) \\ -R_m^{-2}\left(\begin{array}{l} 0.090\sin(5\theta) + 0.106\sin(7\theta) - 2.745\cos(4\theta) \\ -0.652\cos(6\theta) + 0.088\cos(8\theta) \end{array}\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\sin(3\theta) + 4.768\sin(5\theta) + 0.823\sin(7\theta) \\ -0.071\sin(9\theta) - 2.316\cos(6\theta) \end{array}\right) \\ -R_m^{-4}\left(\begin{array}{l} 1.558\sin(5\theta) + 2.180\sin(7\theta) - 0.227\sin(9\theta) \\ +1.235\cos(4\theta) - 0.530\cos(8\theta) + 0.016\cos(10\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1112} = \left[\begin{array}{l} R_m^{-2}\left(1.098\cos(4\theta) + 0.052\cos(6\theta)\right) \\ -R_m^{-3}\left(0.260\cos(6\theta) + 0.013\cos(8\theta)\right) \\ -R_m^{-4}\left(0.165\cos(4\theta) - 0.046\cos(8\theta)\right) \end{array}\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} -0.016\cos(6\theta) + 0.005\cos(8\theta) \\ +R_m^{-2} \left(\begin{array}{l} 0.090\sin(5\theta) + 0.106\sin(7\theta) + 2.745\cos(4\theta) \\ +0.652\cos(6\theta) - 0.088\cos(8\theta) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373\sin(3\theta) + 4.768\sin(5\theta) + 0.823\sin(7\theta) \\ -0.071\sin(9\theta) + 2.316\cos(6\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 1.558\sin(5\theta) + 2.180\sin(7\theta) - 0.227\sin(9\theta) \\ -1.235\cos(4\theta) + 0.530\cos(8\theta) - 0.016\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} 0.053R_m^{-1}\cos(7\theta) \\ -R_m^{-2} \left(\begin{array}{l} 2.745\left(\sin(2\theta) - \sin(4\theta)\right) + 0.033\sin(8\theta) \\ +0.260\cos(7\theta) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 0.057\sin(6\theta) - 1.373\cos(3\theta) + 4.192\cos(5\theta) \\ -0.073\cos(9\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.269\sin(6\theta) - 0.384\sin(8\theta) + 0.016\sin(10\theta) \\ -0.351\cos(5\theta) - 1.606\cos(7\theta) + 0.232\cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[-1.098R_m^{-2}\left(\sin(2\theta) - \sin(4\theta)\right) - R_m^{-4}\left(0.324\sin(6\theta) - 0.017\sin(8\theta)\right)\right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} -0.053 R_m^{-1} \cos(7\theta) \\ -R_m^{-2} \left(\begin{array}{l} 2.745 (\sin(2\theta) - \sin(4\theta)) + 0.033 \sin(8\theta) \\ -0.260 \cos(7\theta) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 0.057 \sin(6\theta) + 1.373 \cos(3\theta) - 4.192 \cos(5\theta) \\ +0.073 \cos(9\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.269 \sin(6\theta) - 0.384 \sin(8\theta) + 0.016 \sin(10\theta) \\ +0.351 \cos(5\theta) + 1.606 \cos(7\theta) - 0.232 \cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

K.4 $\phi = 90,\ 3 < R_m < 15$

$$s_{1001} = \left[\begin{array}{l} R_m^{-2} (0.024 \sin(5\theta) - 5.491 \cos(2\theta) - 2.745 \cos(4\theta)) \\ -R_m^{-3} (6.864 \sin(3\theta) + 4.502 \sin(5\theta) + 0.111 \sin(7\theta)) \\ +R_m^{-4} \left(\begin{array}{l} 1.886 \sin(5\theta) + 0.964 \sin(7\theta) + 3.706 \cos(4\theta) \\ +0.234 \cos(6\theta) - 0.092 \cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[\begin{array}{l} -R_m^{-2} (2.196 \cos(2\theta) + 1.098 \cos(4\theta)) \\ +R_m^{-4} (0.494 \cos(4\theta) + 0.319 \cos(6\theta)) \end{array} \right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} -R_m^{-2} (0.024 \sin(5\theta) + 5.491 \cos(2\theta) + 2.745 \cos(4\theta)) \\ +R_m^{-3} (6.864 \sin(3\theta) + 4.502 \sin(5\theta) + 0.111 \sin(7\theta)) \\ -R_m^{-4} \left(\begin{array}{l} 1.886 \sin(5\theta) + 0.964 \sin(7\theta) - 3.706 \cos(4\theta) \\ -0.234 \cos(6\theta) + 0.092 \cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} -R_m^{-2} (2.745 (\sin(2\theta) + \sin(4\theta)) + 0.017 \cos(5\theta)) \\ +R_m^{-3} \left(\begin{array}{l} 0.034 \sin(6\theta) + 4.118 \cos(3\theta) + 4.393 \cos(5\theta) \\ +0.114 \cos(7\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.471 \sin(4\theta) + 2.255 \sin(6\theta) - 0.092 \sin(8\theta) \\ -1.347 \cos(5\theta) - 0.980 \cos(7\theta) + 0.019 \cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = [-1.098 R_m^{-2} (\sin(2\theta) + \sin(4\theta)) + R_m^{-4} (0.329 \sin(4\theta) + 0.322 \sin(6\theta))] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} -R_m^{-2}\left(2.745\left(\sin\left(2\theta\right)+\sin\left(4\theta\right)\right)-0.017\cos\left(5\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 0.034\sin\left(6\theta\right)-4.118\cos\left(3\theta\right)-4.393\cos\left(5\theta\right) \\ -0.114\cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 2.471\sin\left(4\theta\right)+2.255\sin\left(6\theta\right)-0.092\sin\left(8\theta\right) \\ +1.347\cos\left(5\theta\right)+0.980\cos\left(7\theta\right)-0.019\cos\left(9\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -R_m^{-2}\left(0.016\sin\left(7\theta\right)-2.745\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\sin\left(3\theta\right)+4.192\sin\left(5\theta\right)+0.255\sin\left(7\theta\right) \\ -0.023\cos\left(6\theta\right)-0.033\cos\left(8\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 0.620\sin\left(5\theta\right)+1.282\sin\left(7\theta\right)-0.020\sin\left(9\theta\right) \\ +1.235\cos\left(4\theta\right)+2.327\cos\left(6\theta\right)-0.209\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1112} = \left[1.098R_m^{-2}\cos\left(4\theta\right) - R_m^{-4}\left(0.165\cos\left(4\theta\right)+0.324\cos\left(6\theta\right)\right)\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} R_m^{-2}\left(0.016\sin\left(7\theta\right)+2.745\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\sin\left(3\theta\right)+4.192\sin\left(5\theta\right)+0.255\sin\left(7\theta\right) \\ +0.023\cos\left(6\theta\right)+0.033\cos\left(8\theta\right) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 0.620\sin\left(5\theta\right)+1.282\sin\left(7\theta\right)-0.020\sin\left(9\theta\right) \\ -1.235\cos\left(4\theta\right)-2.327\cos\left(6\theta\right)+0.209\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} -2.745 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) \\ -R_m^{-3} \left(\begin{array}{l} 0.034 \sin\left(8\theta\right) - 1.373 \cos\left(3\theta\right) + 4.143 \cos\left(3\theta\right) \\ +0.120 \cos\left(7\theta\right) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.439 \sin\left(6\theta\right) - 0.211 \sin\left(8\theta\right) - 0.207 \cos\left(5\theta\right) \\ -1.010 \cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[-1.098 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) - 0.327 R_m^{-4} \sin\left(6\theta\right)\right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} -2.745 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) \\ -R_m^{-3} \left(\begin{array}{l} 0.034 \sin\left(8\theta\right) + 1.373 \cos\left(3\theta\right) - 4.143 \cos\left(3\theta\right) \\ -0.120 \cos\left(7\theta\right) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.439 \sin\left(6\theta\right) - 0.211 \sin\left(8\theta\right) + 0.207 \cos\left(5\theta\right) \\ +1.010 \cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

K.5 $\phi = 180,\ 2 < R_m < 3$

$$s_{1001} = \left[\begin{array}{l} -R_m^{-2} \left(2.745 \left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) - 0.061 \sin\left(5\theta\right) - 0.124 \sin\left(7\theta\right)\right) \\ +R_m^{-3} \left(\begin{array}{l} 1.373 \sin\left(3\theta\right) + 3.896 \sin\left(5\theta\right) - 0.057 \sin\left(6\theta\right) \\ -0.927 \sin\left(7\theta\right) + 0.159 \sin\left(8\theta\right) - 0.073 \sin\left(9\theta\right) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.269 \sin\left(6\theta\right) - 2.347 \sin\left(7\theta\right) + 0.572 \sin\left(8\theta\right) \\ -0.232 \sin\left(9\theta\right) + 0.016 \sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[1.098 R_m^{-2} \left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) - R_m^{-4} \left(0.324 \sin\left(6\theta\right) + 0.017 \sin\left(8\theta\right)\right)\right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} -R_m^{-2} \left(2.745 \left(\sin\left(2\theta\right) + \sin\left(4\theta\right)\right) + 0.061 \sin\left(5\theta\right) + 0.124 \sin\left(7\theta\right)\right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373 \sin\left(3\theta\right) + 3.896 \sin\left(5\theta\right) + 0.057 \sin\left(6\theta\right) \\ -0.927 \sin\left(7\theta\right) - 0.159 \sin\left(8\theta\right) - 0.073 \sin\left(9\theta\right) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.269 \sin\left(6\theta\right) + 2.347 \sin\left(7\theta\right) + 0.572 \sin\left(8\theta\right) \\ +0.232 \sin\left(9\theta\right) + 0.016 \sin\left(10\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} -0.016\cos(6\theta) - 0.047R_m^{-1}\cos(7\theta) \\ +R_m^{-2}\left(\begin{array}{l} 2.745\cos(4\theta) + 0.090\cos(5\theta) - 0.652\cos(6\theta) \\ +0.234\cos(7\theta) + 0.085\cos(8\theta) \end{array} \right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\cos(3\theta) - 4.768\cos(5\theta) + 2.316\cos(6\theta) \\ -0.557\cos(8\theta) + 0.071\cos(9\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 1.235\cos(4\theta) - 1.558\cos(5\theta) + 1.523\cos(7\theta) \\ -1.031\cos(8\theta) + 0.227\cos(9\theta) - 0.016\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = \left[\begin{array}{l} 1.098R_m^{-2}\cos(4\theta) - 0.013R_m^{-3}\cos(8\theta) \\ +R_m^{-4}\left(0.165\cos(4\theta) - 0.318\cos(6\theta) - 0.046\cos(8\theta)\right) \end{array} \right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} -0.016\cos(6\theta) + 0.047R_m^{-1}\cos(7\theta) \\ +R_m^{-2}\left(\begin{array}{l} 2.745\cos(4\theta) - 0.090\cos(5\theta) - 0.652\cos(6\theta) \\ -0.234\cos(7\theta) + 0.085\cos(8\theta) \end{array} \right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\cos(3\theta) - 4.768\cos(5\theta) - 2.316\cos(6\theta) \\ +0.557\cos(8\theta) + 0.071\cos(9\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 1.235\cos(4\theta) + 1.558\cos(5\theta) - 1.523\cos(7\theta) \\ -1.031\cos(8\theta) - 0.227\cos(9\theta) - 0.016\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} 0.093R_m^{-1}\sin(6\theta) \\ -R_m^{-2}\left(\begin{array}{l} 2.745\left(\sin(2\theta) - \sin(4\theta)\right) - 0.149\sin(5\theta) \\ +0.776\sin(6\theta) - 0.003\sin(10\theta) \end{array} \right) \\ +R_m^{-3}\left(\begin{array}{l} 4.118\sin(3\theta) - 5.202\sin(5\theta) + 2.278\sin(6\theta) \\ +0.293\sin(7\theta) - 0.147\sin(8\theta) + 0.069\sin(9\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 2.597\sin(5\theta) + 1.512\sin(7\theta) \\ -0.538\sin(8\theta) + 0.222\sin(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1112} = \left[\begin{array}{l} -0.023R_m^{-1}\sin(4\theta) \\ -R_m^{-2}\left(1.098\sin(2\theta) - 1.269\sin(4\theta) + 0.049\sin(6\theta)\right) \\ -R_m^{-3}\left(0.413\sin(4\theta) - 0.251\sin(6\theta)\right) + 0.017R_m^{-4}\sin(8\theta) \end{array} \right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} 0.093R_m^{-1}\sin(6\theta) \\ -R_m^{-2}\left(\begin{array}{l} 2.745\left(\sin(2\theta) - \sin(4\theta)\right) + 0.149\sin(5\theta) \\ +0.776\sin(6\theta) - 0.003\sin(10\theta) \end{array}\right) \\ -R_m^{-3}\left(\begin{array}{l} 4.118\sin(3\theta) - 5.202\sin(5\theta) - 2.278\sin(6\theta) \\ +0.293\sin(7\theta) + 0.147\sin(8\theta) + 0.069\sin(9\theta) \end{array}\right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) + 2.597\sin(5\theta) - 1.512\sin(7\theta) \\ -0.538\sin(8\theta) - 0.222\sin(9\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} -0.015R_m^{-1}\cos(7\theta) \\ +R_m^{-2}\left(\begin{array}{l} 5.491\cos(2\theta) - 2.745\cos(4\theta) - 0.209\cos(5\theta) \\ +0.294\cos(6\theta) \end{array}\right) \\ -R_m^{-3}\left(\begin{array}{l} 6.864\cos(3\theta) - 5.635\cos(5\theta) + 1.633\cos(6\theta) \\ -0.141\cos(8\theta) + 0.067\cos(9\theta) \end{array}\right) \\ +R_m^{-4}\left(\begin{array}{l} 3.706\cos(4\theta) - 3.636\cos(5\theta) + 1.013\cos(7\theta) \\ -0.520\cos(8\theta) + 0.217\cos(9\theta) - 0.015\cos(10\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{2112} = \left[\begin{array}{l} R_m^{-2}\left(2.196\cos(2\theta) - 1.098\cos(4\theta) + 0.047\cos(6\theta)\right) \\ -0.242R_m^{-3}\cos(6\theta) + R_m^{-4}\left(0.494\cos(4\theta) - 0.017\cos(8\theta)\right) \end{array}\right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} 0.015R_m^{-1}\cos(7\theta) \\ +R_m^{-2}\left(\begin{array}{l} 5.491\cos(2\theta) - 2.745\cos(4\theta) - 0.209\cos(5\theta) \\ +0.294\cos(6\theta) \end{array} \right) \\ +R_m^{-3}\left(\begin{array}{l} 6.864\cos(3\theta) - 5.635\cos(5\theta) - 1.633\cos(6\theta) \\ +0.141\cos(8\theta) + 0.067\cos(9\theta) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 3.706\cos(4\theta) - 3.636\cos(5\theta) + 1.013\cos(7\theta) \\ -0.520\cos(8\theta) + 0.217\cos(9\theta) - 0.015\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

K.6 $\phi = 180,\ 3 < R_m < 15$

$$s_{1001} = \left[\begin{array}{l} -2.745R_m^{-2}\left(\sin(2\theta) + \sin(4\theta)\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\sin(3\theta) + 4.142\sin(5\theta) - 0.120\sin(7\theta) \\ +0.034\sin(8\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 0.207\sin(5\theta) + 2.439\sin(6\theta) - 1.010\sin(7\theta) \\ +0.211\sin(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[-1.098R_m^{-2}\left(\sin(2\theta) + \sin(4\theta)\right) - 0.327R_m^{-4}\sin(6\theta)\right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} -2.745R_m^{-2}\left(\sin(2\theta) + \sin(4\theta)\right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\sin(3\theta) + 4.142\sin(5\theta) - 0.120\sin(7\theta) \\ -0.034\sin(8\theta) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 0.207\sin(5\theta) - 2.439\sin(6\theta) - 1.010\sin(7\theta) \\ -0.211\sin(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} R_m^{-2}\left(2.745\cos(4\theta) - 0.016\cos(7\theta)\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\cos(3\theta) - 4.192\cos(5\theta) + 0.023\cos(6\theta) \\ +0.255\cos(7\theta) - 0.033\cos(8\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 1.235\cos(4\theta) - 0.620\cos(5\theta) - 2.327\cos(6\theta) \\ +1.282\cos(7\theta) - 0.209\cos(8\theta) + 0.020\cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = \left[1.098R_m^{-2}\cos(4\theta) - R_m^{-4}\left(0.165\cos(4\theta) - 0.324\cos(6\theta)\right)\right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} R_m^{-2}\left(2.745\cos\left(4\theta\right) + 0.016\cos\left(7\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\cos\left(3\theta\right) - 4.192\cos\left(5\theta\right) - 0.023\cos\left(6\theta\right) \\ +0.255\cos\left(7\theta\right) + 0.033\cos\left(8\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 1.235\cos\left(4\theta\right) + 0.620\cos\left(5\theta\right) - 2.327\cos\left(6\theta\right) \\ -1.282\cos\left(7\theta\right) - 0.209\cos\left(8\theta\right) - 0.020\cos\left(9\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -R_m^{-2}\left(2.745\left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) - 0.017\sin\left(5\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 4.118\sin\left(3\theta\right) - 4.393\sin\left(5\theta\right) + 0.034\sin\left(6\theta\right) \\ +0.114\sin\left(7\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin\left(4\theta\right) - 1.347\sin\left(5\theta\right) - 2.255\sin\left(6\theta\right) \\ +0.980\sin\left(7\theta\right) - 0.092\sin\left(8\theta\right) + 0.019\sin\left(9\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1112} = \left[-1.098R_m^{-2}\left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) - R_m^{-4}\left(0.393\sin\left(4\theta\right) - 0.322\sin\left(6\theta\right)\right)\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} -R_m^{-2}\left(2.745\left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) + 0.017\sin\left(5\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 4.118\sin\left(3\theta\right) - 4.393\sin\left(5\theta\right) - 0.034\sin\left(6\theta\right) \\ +0.114\sin\left(7\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin\left(4\theta\right) + 1.347\sin\left(5\theta\right) - 2.255\sin\left(6\theta\right) \\ -0.980\sin\left(7\theta\right) - 0.092\sin\left(8\theta\right) - 0.019\sin\left(9\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} \cos(6\theta)\,(0.0014 - 0.040R_m^{-1}) \\ +R_m^{-2}\,(5.491\cos(2\theta) - 2.745\cos(4\theta) + 0.394\cos(6\theta)) \\ -R_m^{-3} \left(\begin{array}{l} 6.864\cos(3\theta) - 4.291\cos(5\theta) + 1.624\cos(6\theta) \\ +0.111\cos(7\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 3.706\cos(4\theta) - 1.447\cos(5\theta) + 0.964\cos(7\theta) \\ -0.092\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[\begin{array}{l} R_m^{-2}\,(2.196\cos(2\theta) - 1.098\cos(4\theta)) \\ +R_m^{-4}\,(0.494\cos(4\theta) - 0.319\cos(6\theta)) \end{array} \right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} \cos(6\theta)\,(0.0014 - 0.040R_m^{-1}) \\ +R_m^{-2}\,(5.491\cos(2\theta) - 2.745\cos(4\theta) + 0.394\cos(6\theta)) \\ +R_m^{-3} \left(\begin{array}{l} 6.864\cos(3\theta) - 4.291\cos(5\theta) - 1.624\cos(6\theta) \\ +0.111\cos(7\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 3.706\cos(4\theta) + 1.447\cos(5\theta) - 0.964\cos(7\theta) \\ -0.092\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

K.7 $\phi = 270,\ 2 < R_m < 3$

$$s_{1001} = \left[\begin{array}{l} R_m^{-2}\,(5.491\cos(2\theta) + 2.745\cos(4\theta) + 0.294\cos(6\theta)) \\ -R_m^{-3} \left(\begin{array}{l} 6.864\sin(3\theta) + 4.637\sin(5\theta) + 0.271\sin(7\theta) \\ -0.067\sin(9\theta) + 1.633\cos(6\theta) + 0.141\cos(8\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.460\sin(5\theta) + 1.442\sin(7\theta) - 0.217\sin(9\theta) \\ -3.706\cos(4\theta) + 0.520\cos(8\theta) - 0.015\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[\begin{array}{l} R_m^{-2}\,(2.196\cos(2\theta) + 1.098\cos(4\theta)) \\ -R_m^{-4}\,(0.494\cos(4\theta) + 0.306\cos(6\theta) - 0.017\cos(8\theta)) \end{array} \right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} R_m^{-2}\,(5.491\cos(2\theta) + 2.745\cos(4\theta) + 0.294\cos(6\theta)) \\ +R_m^{-3} \left(\begin{array}{l} 6.864\sin(3\theta) + 4.637\sin(5\theta) + 0.271\sin(7\theta) \\ 0.067\sin(9\theta) - 1.633\cos(6\theta) - 0.141\cos(8\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.460\sin(5\theta) + 1.442\sin(7\theta) - 0.217\sin(9\theta) \\ +3.706\cos(4\theta) - 0.520\cos(8\theta) + 0.015\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} 0.030 \ln R_m^2 \cos(5\theta) - R_m^{-1}(0.093\sin(6\theta) + 0.772\cos(5\theta)) \\ +R_m^{-2}\left(\begin{array}{l} 2.745(\sin(2\theta) + \sin(4\theta)) + 0.776\sin(6\theta) \\ +3.287\cos(5\theta) - 0.038\cos(9\theta) \end{array}\right) \\ -R_m^{-3}\left(\begin{array}{l} 2.278\sin(6\theta) + 0.147\sin(8\theta) - 4.118\cos(3\theta) \\ -0.293\cos(7\theta) - 0.117\cos(9\theta) \end{array}\right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 0.538\sin(8\theta) + 0.016\sin(10\theta) \\ +1.512\cos(7\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1012} = \left[\begin{array}{l} -0.023 R_m^{-1}\sin(4\theta) \\ +R_m^{-2}(1.098\sin(2\theta) + 1.269\sin(4\theta) + 0.049\sin(6\theta)) \\ -R_m^{-3}(0.413\sin(4\theta) + 0.251\sin(6\theta)) + 0.017 R_m^{-4}\sin(8\theta) \end{array}\right] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} -0.030 \ln R_m^2 \cos(5\theta) - R_m^{-1}(0.093\sin(6\theta) - 0.772\cos(5\theta)) \\ +R_m^{-2}\left(\begin{array}{l} 2.745(\sin(2\theta) + \sin(4\theta)) + 0.776\sin(6\theta) \\ -3.287\cos(5\theta) + 0.038\cos(9\theta) \end{array}\right) \\ -R_m^{-3}\left(\begin{array}{l} 2.278\sin(6\theta) + 0.147\sin(8\theta) + 4.118\cos(3\theta) \\ +0.293\cos(7\theta) + 0.117\cos(9\theta) \end{array}\right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin(4\theta) - 0.538\sin(8\theta) + 0.016\sin(10\theta) \\ -1.512\cos(7\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -R_m^{-2}\left(\begin{array}{l} 0.090\sin(5\theta) + 0.106\sin(7\theta) + 2.745\cos(4\theta) \\ +0.085\cos(8\theta) \end{array}\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\sin(3\theta) + 4.768\sin(5\theta) + 0.823\sin(7\theta) \\ -0.071\sin(9\theta) + 0.114\cos(6\theta) + 0.557\cos(8\theta) \end{array}\right) \\ -R_m^{-4}\left(\begin{array}{l} 1.558\sin(5\theta) + 2.180\sin(7\theta) - 0.227\sin(9\theta) \\ -1.235\cos(4\theta) - 2.066\cos(6\theta) + 1.031\cos(8\theta) \\ -0.016\cos(10\theta) \end{array}\right) \end{array}\right] \times 10^3$$

$$s_{1112} = \left[\begin{array}{l} -R_m^{-2}(1.098\cos(4\theta) - 0.003\cos(8\theta)) \\ +R_m^{-4}(0.165\cos(4\theta) + 0.318\cos(6\theta) - 0.031\cos(8\theta)) \end{array}\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} R_m^{-2} \left(\begin{array}{l} 0.090\sin(5\theta) + 0.106\sin(7\theta) - 2.745\cos(4\theta) \\ -0.085\cos(8\theta) \end{array} \right) \\ -R_m^{-3} \left(\begin{array}{l} 1.373\sin(3\theta) + 4.768\sin(5\theta) + 0.823\sin(7\theta) \\ -0.071\sin(9\theta) - 0.114\cos(6\theta) - 0.557\cos(8\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 1.558\sin(5\theta) + 2.180\sin(7\theta) - 0.227\sin(9\theta) \\ +1.235\cos(4\theta) + 2.066\cos(6\theta) - 1.031\cos(8\theta) \\ +0.016\cos(10\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} 0.038\cos(5\theta) \\ +R_m^{-2} \left(\begin{array}{l} 2.745\left(\sin(2\theta) - \sin(4\theta)\right) - 0.011\sin(10\theta) \\ -1.318\cos(5\theta) + 0.124\cos(7\theta) + 0.040\cos(9\theta) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 0.057\sin(6\theta) + 0.159\sin(8\theta) + 0.033\sin(10\theta) \\ +1.373\cos(3\theta) - 0.927\cos(7\theta) - 0.122\cos(9\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.269\sin(6\theta) - 0.572\sin(8\theta) - 3.370\cos(5\theta) \\ +2.347\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[1.098R_m^{-2}\left(\sin(2\theta) - \sin(4\theta)\right) + R_m^{-4}\left(0.324\sin(6\theta) - 0.017\sin(8\theta)\right)\right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} -0.038\cos(5\theta) \\ +R_m^{-2} \left(\begin{array}{l} 2.745(\sin(2\theta) - \sin(4\theta)) - 0.011\sin(10\theta) \\ +1.318\cos(5\theta) - 0.124\cos(7\theta) - 0.040\cos(9\theta) \end{array} \right) \\ +R_m^{-3} \left(\begin{array}{l} 0.057\sin(6\theta) + 0.159\sin(8\theta) + 0.033\sin(10\theta) \\ -1.373\cos(3\theta) + 0.927\cos(7\theta) + 0.122\cos(9\theta) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.269\sin(6\theta) - 0.572\sin(8\theta) + 3.370\cos(5\theta) \\ -2.347\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^3$$

K.8 $\phi = 270,\ 3 < R_m < 15$

$$s_{1001} = \left[\begin{array}{l} R_m^{-2}(0.024\sin(5\theta) + 5.491\cos(2\theta) + 2.745\cos(4\theta)) \\ -R_m^{-3}(6.864\sin(3\theta) + 4.502\sin(5\theta) + 0.111\sin(7\theta)) \\ +R_m^{-4} \left(\begin{array}{l} 1.886\sin(5\theta) + 0.964\sin(7\theta) - 3.706\cos(4\theta) \\ -2.343\cos(6\theta) + 0.092\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1002} = \left[\begin{array}{l} R_m^{-2}(2.196\cos(2\theta) + 1.098\cos(4\theta)) \\ -R_m^{-4}(0.494\cos(4\theta) + 0.319\cos(6\theta)) \end{array} \right] \times 10^4$$

$$s_{1003} = \left[\begin{array}{l} -R_m^{-2}(0.024\sin(5\theta) - 5.491\cos(2\theta) - 2.745\cos(4\theta)) \\ +R_m^{-3}(6.864\sin(3\theta) + 4.502\sin(5\theta) + 0.111\sin(7\theta)) \\ -R_m^{-4} \left(\begin{array}{l} 1.886\sin(5\theta) + 0.964\sin(7\theta) + 3.706\cos(4\theta) \\ +2.343\cos(6\theta) - 0.092\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1011} = \left[\begin{array}{l} R_m^{-2}(2.745(\sin(2\theta) + \sin(4\theta)) - 0.017\cos(5\theta)) \\ -R_m^{-3} \left(\begin{array}{l} 0.034\sin(6\theta) - 4.118\cos(3\theta) - 4.393\cos(5\theta) \\ -0.114\cos(7\theta) \end{array} \right) \\ -R_m^{-4} \left(\begin{array}{l} 2.471\sin(4\theta) + 2.255\sin(6\theta) - 0.092\sin(8\theta) \\ +1.347\cos(5\theta) + 0.980\cos(7\theta) - 0.019\cos(9\theta) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1012} = [1.098R_m^{-2}(\sin(2\theta) + \sin(4\theta)) - R_m^{-4}(0.329\sin(4\theta) - 0.322\sin(6\theta))] \times 10^4$$

$$s_{1013} = \left[\begin{array}{l} R_m^{-2}\left(2.745\left(\sin\left(2\theta\right)+\sin\left(4\theta\right)\right)+0.017\cos\left(5\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 0.034\sin\left(6\theta\right)+4.118\cos\left(3\theta\right)+4.393\cos\left(5\theta\right) \\ +0.114\cos\left(7\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 2.471\sin\left(4\theta\right)+2.255\sin\left(6\theta\right)-0.092\sin\left(8\theta\right) \\ -1.347\cos\left(5\theta\right)-0.980\cos\left(7\theta\right)+0.019\cos\left(9\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1111} = \left[\begin{array}{l} -R_m^{-2}\left(0.016\sin\left(7\theta\right)+2.745\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 1.373\sin\left(3\theta\right)+4.192\sin\left(5\theta\right)+0.255\sin\left(7\theta\right) \\ +0.023\cos\left(6\theta\right)+0.033\cos\left(8\theta\right) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 0.620\sin\left(5\theta\right)+1.282\sin\left(7\theta\right)-0.020\sin\left(9\theta\right) \\ -1.235\cos\left(4\theta\right)-2.327\cos\left(6\theta\right)+0.209\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{1112} = \left[-1.098R_m^{-2}\cos\left(4\theta\right)+R_m^{-4}\left(0.165\cos\left(4\theta\right)+0.324\cos\left(6\theta\right)\right)\right] \times 10^4$$

$$s_{1113} = \left[\begin{array}{l} R_m^{-2}\left(0.016\sin\left(7\theta\right)-2.745\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 1.373\sin\left(3\theta\right)+4.192\sin\left(5\theta\right)+0.255\sin\left(7\theta\right) \\ -0.023\cos\left(6\theta\right)-0.033\cos\left(8\theta\right) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 0.620\sin\left(5\theta\right)+1.282\sin\left(7\theta\right)-0.020\sin\left(9\theta\right) \\ +1.235\cos\left(4\theta\right)+2.327\cos\left(6\theta\right)-0.209\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2001} = s_{1011}$$

$$s_{2002} = s_{1012}$$

$$s_{2003} = s_{1013}$$

$$s_{2011} = s_{1111}$$

$$s_{2012} = s_{1112}$$

$$s_{2013} = s_{1113}$$

$$s_{2111} = \left[\begin{array}{l} 2.745 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) \\ +R_m^{-3} \left(\begin{array}{l} 0.034 \sin\left(8\theta\right) + 1.373 \cos\left(3\theta\right) - 4.143 \cos\left(5\theta\right) \\ -0.120 \cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.439 \sin\left(6\theta\right) - 0.211 \sin\left(8\theta\right) + 0.207 \cos\left(5\theta\right) \\ +1.010 \cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

$$s_{2112} = \left[1.098 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) + 0.327 R_m^{-4} \sin\left(6\theta\right)\right] \times 10^4$$

$$s_{2113} = \left[\begin{array}{l} 2.745 R_m^{-2} \left(\sin\left(2\theta\right) - \sin\left(4\theta\right)\right) \\ +R_m^{-3} \left(\begin{array}{l} 0.034 \sin\left(8\theta\right) - 1.373 \cos\left(3\theta\right) + 4.143 \cos\left(5\theta\right) \\ +0.120 \cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-4} \left(\begin{array}{l} 2.439 \sin\left(6\theta\right) - 0.211 \sin\left(8\theta\right) - 0.207 \cos\left(5\theta\right) \\ -1.010 \cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^3$$

APPENDIX L

Surface Fit Equations - d terms

The notation used in this section is d_{lijk} where l, i and j are the tensor terms and k is the node number on the field element.

L.1 $\phi = 0,\ 2 < R_m < 3$

$$d_{1001} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.182\cos\left(2\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.875\cos\left(4\theta\right) - 0.044\cos\left(6\theta\right)\right) \\ +R_m^{-3}\left(\begin{array}{l} 3.194\cos\left(2\theta\right) + 0.269\cos\left(3\theta\right) + 0.388\cos\left(5\theta\right) \\ +0.215\cos\left(6\theta\right) + 0.022\cos\left(7\theta\right) \end{array} \right) \\ -2.544R_m^{-4}\cos\left(2\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1002} = \left[\begin{array}{l} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.339\cos\left(3\theta\right)\right) + 0.030R_m^{-2}\cos\left(3\theta\right) \\ +0.052R_m^{-3}\cos\left(5\theta\right) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) - 0.182\cos\left(2\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.875\cos\left(4\theta\right) - 0.044\cos\left(6\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 3.194\cos\left(2\theta\right) - 0.269\cos\left(3\theta\right) - 0.388\cos\left(5\theta\right) \\ +0.215\cos\left(6\theta\right) - 0.022\cos\left(7\theta\right) \end{array} \right) \\ +2.544R_m^{-4}\cos\left(2\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) + 0.379R_m^{-3}\sin\left(5\theta\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) - 0.050\sin\left(7\theta\right) \\ -0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{1012} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.451\sin\left(3\theta\right)\right) + 0.510R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) + 0.379R_m^{-3}\sin\left(5\theta\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.060\sin\left(4\theta\right) + 0.252\sin\left(6\theta\right) - 0.050\sin\left(7\theta\right) \\ +0.014\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right) + 0.158\cos\left(7\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right) - 0.578\cos\left(7\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.890\cos\left(4\theta\right) + 0.773\cos\left(5\theta\right) - 2.471\cos\left(6\theta\right) \\ -0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.503\cos\left(5\theta\right)\right) - 0.040R_m^{-4}\cos\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right) - 0.158\cos\left(7\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right) - 0.578\cos\left(7\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 1.890\cos\left(4\theta\right) - 0.773\cos\left(5\theta\right) - 2.471\cos\left(6\theta\right) \\ -0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) - 8.971\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) + 1.680\sin\left(4\theta\right) - 4.049\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.803\sin\left(5\theta\right) - 2.468\sin\left(6\theta\right) - 0.492\sin\left(7\theta\right) \\ -0.133\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin(\theta) - 3.510\sin(3\theta) - 0.030\sin(5\theta)\right) \\ +0.299R_m^{-2}\sin(3\theta) + R_m^{-4}\left(0.772\sin(5\theta) + 0.040\sin(7\theta)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin(\theta) - 8.621\sin(3\theta)\right) \\ +R_m^{-2}\left(4.642\sin(2\theta) - 8.971\sin(4\theta)\right) \\ -R_m^{-3}\left(2.686\sin(3\theta) - 1.680\sin(4\theta) - 4.049\sin(5\theta)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.803\sin(5\theta) + 2.468\sin(6\theta) - 0.492\sin(7\theta) \\ +0.133\sin(8\theta) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos(\theta) - 0.862\cos(3\theta)\right) \\ +R_m^{-2}\left(1.326\cos(2\theta) - 0.862\cos(4\theta)\right) \\ +R_m^{-3}\left(0.527\cos(3\theta) - 0.413\cos(5\theta) - 0.021\cos(7\theta)\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.328\cos(4\theta) + 0.120\cos(5\theta) - 0.242\cos(6\theta) \\ -0.013\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{2012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\cos(\theta) - 3.569\cos(3\theta)\right) + 0.587R_m^{-2}\cos(3\theta) \\ -0.495R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-2}$$

$$d_{2013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos(\theta) - 0.862\cos(3\theta)\right) \\ -R_m^{-2}\left(1.326\cos(2\theta) - 0.862\cos(4\theta)\right) \\ +R_m^{-3}\left(0.527\cos(3\theta) - 0.413\cos(5\theta) - 0.021\cos(7\theta)\right) \\ -R_m^{-4}\left(\begin{array}{l} 0.328\cos(4\theta) - 0.120\cos(5\theta) - 0.242\cos(6\theta) \\ -0.013\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{2111} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin(\theta) - 0.862\sin(3\theta)\right) \\ +R_m^{-2}\left(2.188\sin(2\theta) - 0.862\sin(4\theta)\right) \\ +R_m^{-3}\left(0.786\sin(3\theta) - 0.352\sin(5\theta)\right) \\ +R_m^{-4}\left(0.458\sin(4\theta) - 0.237\sin(6\theta) - 0.048\sin(7\theta)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin(\theta) - 0.345\sin(3\theta) + 0.008\sin(5\theta)\right) \\ -0.039R_m^{-2}\sin(5\theta) + 0.105R_m^{-3}\sin(3\theta) \end{array} \right] \times 10^{-1}$$

$$d_{2113} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.352\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.237\sin\left(6\theta\right) - 0.048\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

L.2 $\phi = 0,\ 3 < R_m < 15$

$$d_{1001} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.867\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.269\cos\left(3\theta\right) + 0.388\cos\left(5\theta\right)\right) + 0.258R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1002} = \begin{bmatrix} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.036\cos\left(3\theta\right) + 0.052\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-1}$$

$$d_{1003} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.867\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.269\cos\left(3\theta\right) + 0.388\cos\left(5\theta\right)\right) - 0.258R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1011} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1012} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.449\sin\left(3\theta\right)\right) + 0.514R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{1013} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1111} = \begin{bmatrix} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{1112} = \begin{bmatrix} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.511\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1113} = \begin{bmatrix} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{2001} = \begin{bmatrix} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) - 3.804\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.989\sin\left(4\theta\right) - 2.535\sin\left(6\theta\right) - 0.330\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{2002} = \begin{bmatrix} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.358\sin\left(3\theta\right) - 0.511\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2003} = \begin{bmatrix} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) - 3.804\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.989\sin\left(4\theta\right) - 2.535\sin\left(6\theta\right) + 0.330\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{2011} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2012} = \begin{bmatrix} R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.703\cos\left(3\theta\right) - 0.508\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2013} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2111} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.105\sin\left(3\theta\right) - 0.051\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$d_{2113} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

L.3 $\phi = 90,\ 2 < R_m < 3$

$$d_{1001} = \left[\begin{array}{l} R_m^{-1}\left(0.007\sin\left(6\theta\right) + 3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.144\sin\left(6\theta\right) + 0.786\cos\left(3\theta\right) + 0.352\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.048\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1002} = \left[\begin{array}{l} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.105\cos\left(3\theta\right) + 0.049\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} -R_m^{-1}\left(0.007\sin\left(6\theta\right) - 3.515\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.144\sin\left(6\theta\right) - 0.786\cos\left(3\theta\right) - 0.352\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.048\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right) + 0.038\cos\left(6\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right) - 0.193\cos\left(6\theta\right) \\ +0.006\cos\left(8\theta\right) \end{array} \right) \\ +R_m^{-4}\left(0.048\sin\left(7\theta\right) + 0.328\cos\left(4\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.495\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right) + 0.038\cos\left(6\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 0.527\sin\left(3\theta\right) + 0.361\sin\left(5\theta\right) + 0.193\cos\left(6\theta\right) \\ -0.006\cos\left(8\theta\right) \end{array} \right) \\ +R_m^{-4}\left(0.048\sin\left(7\theta\right) - 0.328\cos\left(4\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right) + 0.399\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(2.003\sin\left(6\theta\right) + 2.686\cos\left(3\theta\right) + 4.049\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.989\sin\left(4\theta\right) - 0.133\sin\left(8\theta\right) - 0.803\cos\left(5\theta\right) \\ -0.492\cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.358\cos\left(3\theta\right) + 0.503\cos\left(5\theta\right) - 0.017\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right) + 0.399\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(2.003\sin\left(6\theta\right) + 2.686\cos\left(3\theta\right) + 4.049\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.989\sin\left(4\theta\right) - 0.133\sin\left(8\theta\right) - 0.803\cos\left(5\theta\right) \\ -0.492\cos\left(7\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -0.085\ln R_m \sin\left(5\theta\right) \\ -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right) - 1.164\sin\left(5\theta\right)\right) \\ -R_m^{-2}\left(3.916\sin\left(5\theta\right) + 3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 0.181\sin\left(7\theta\right) - 0.058\cos\left(8\theta\right)\right) \\ +R_m^{-4}\left(0.901\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) + 2.471\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.503\sin\left(5\theta\right)\right) + 0.040R_m^{-4}\sin\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \begin{bmatrix} -0.085 \ln R_m \sin(5\theta) \\ -R_m^{-1}(0.663\sin(\theta) - 8.621\sin(3\theta) - 1.164\sin(5\theta)) \\ -R_m^{-2}(3.916\sin(5\theta) - 3.979\cos(2\theta) - 8.621\cos(4\theta)) \\ -R_m^{-3}(2.487\sin(3\theta) + 0.181\sin(7\theta) + 0.058\cos(8\theta)) \\ +R_m^{-4}(0.901\sin(7\theta) - 1.890\cos(4\theta) - 2.471\cos(6\theta)) \end{bmatrix} \times 10^{-3}$$

$$d_{2011} = \begin{bmatrix} R_m^{-1}(0.064\sin(2\theta) + 1.790\cos(\theta) - 0.864\cos(3\theta)) \\ -0.862R_m^{-2}\cos(4\theta) \\ +R_m^{-3}(1.118\sin(2\theta) + 0.379\cos(5\theta)) \\ -R_m^{-4}\begin{pmatrix} 0.890\sin(2\theta) - 0.060\sin(4\theta) - 0.252\sin(6\theta) \\ +0.014\sin(8\theta) + 0.050\cos(7\theta) \end{pmatrix} \end{bmatrix} \times 10^{-2}$$

$$d_{2012} = [R_m^{-1}(7.162\cos(\theta) - 3.451\cos(3\theta)) + 0.510R_m^{-3}\cos(5\theta)] \times 10^{-2}$$

$$d_{2013} = \begin{bmatrix} -R_m^{-1}(0.064\sin(2\theta) - 1.790\cos(\theta) + 0.864\cos(3\theta)) \\ +0.862R_m^{-2}\cos(4\theta) \\ -R_m^{-3}(1.118\sin(2\theta) - 0.379\cos(5\theta)) \\ -R_m^{-4}\begin{pmatrix} 0.890\sin(2\theta) - 0.060\sin(4\theta) - 0.252\sin(6\theta) \\ +0.014\sin(8\theta) + 0.050\cos(7\theta) \end{pmatrix} \end{bmatrix} \times 10^{-2}$$

$$d_{2111} = \begin{bmatrix} R_m^{-1}(3.515\sin(\theta) - 0.862\sin(3\theta) - 0.182\cos(2\theta)) \\ +0.875R_m^{-2}\cos(4\theta) \\ -R_m^{-3}(0.269\sin(3\theta) - 0.388\sin(5\theta) + 3.194\cos(2\theta)) \\ -R_m^{-4}(0.051\sin(7\theta) - 2.544\cos(2\theta) + 0.256\cos(6\theta)) \end{bmatrix} \times 10^{-2}$$

$$d_{2112} = \begin{bmatrix} R_m^{-1}(1.406\sin(\theta) - 0.345\sin(3\theta) + 0.009\sin(5\theta)) \\ +0.043R_m^{-2}\sin(5\theta) - 0.036R_m^{-3}\sin(3\theta) \end{bmatrix} \times 10^{-1}$$

$$d_{2113} = \begin{bmatrix} R_m^{-1}(3.515\sin(\theta) - 0.862\sin(3\theta) + 0.182\cos(2\theta)) \\ -0.875R_m^{-2}\cos(4\theta) \\ -R_m^{-3}(0.269\sin(3\theta) - 0.388\sin(5\theta) - 3.194\cos(2\theta)) \\ -R_m^{-4}(0.051\sin(7\theta) + 2.544\cos(2\theta) - 0.256\cos(6\theta)) \end{bmatrix} \times 10^{-2}$$

L.4 $\phi = 90,\ 3 < R_m < 15$

$$d_{1001} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1002} = \left[\begin{array}{l} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.344\cos\left(3\theta\right)\right) - 0.051R_m^{-3}\cos\left(5\theta\right) \\ -0.307R_m^{-4}\cos\left(3\theta\right) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.508\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.686\cos\left(3\theta\right) + 3.804\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.358\cos\left(3\theta\right) + 0.511\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.686\cos\left(3\theta\right) + 3.804\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) + 2.537\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.511\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2012} = \left[R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.449\cos\left(3\theta\right)\right) + 0.514R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{2013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2111} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.867\cos\left(4\theta\right) - 1.326\cos\left(2\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right)\right) - 0.258R_m^{-4}\cos\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.036\sin\left(3\theta\right) - 0.052\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$d_{2113} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.867\cos\left(4\theta\right) - 1.326\cos\left(2\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right)\right) + 0.258R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

L.5 $\phi = 180,\ 2 < R_m < 3$

$$d_{1001} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.875\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.269\cos\left(3\theta\right) + 0.388\cos\left(5\theta\right) + 0.022\cos\left(7\theta\right)\right) \\ -0.256R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1002} = \begin{bmatrix} \cos\left(3\theta\right)\left(0.163 - \left(0.067\ln R_m\right)\right) + 1.406R_m^{-1}\cos\left(\theta\right) \\ +R_m^{-2}\left(0.243\cos\left(3\theta\right) + 0.011\cos\left(5\theta\right)\right) + 0.061R_m^{-4}\cos\left(5\theta\right) \end{bmatrix} \times 10^{-1}$$

$$d_{1003} = \begin{bmatrix} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.875\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.269\cos\left(3\theta\right) + 0.388\cos\left(5\theta\right) + 0.022\cos\left(7\theta\right)\right) \\ +0.256R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1011} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.379\sin\left(5\theta\right) - 0.006\sin\left(8\theta\right)\right) \\ +R_m^{-4}\left(0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) + 0.050\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1012} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.451\sin\left(3\theta\right)\right) + 0.510R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{1013} = \begin{bmatrix} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.864\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.379\sin\left(5\theta\right) + 0.006\sin\left(8\theta\right)\right) \\ -R_m^{-4}\left(0.060\sin\left(4\theta\right) - 0.252\sin\left(6\theta\right) - 0.050\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{1111} = \begin{bmatrix} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.471\cos\left(6\theta\right) - 0.133\cos\left(8\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.503\cos\left(5\theta\right)\right) - 0.040R_m^{-4}\cos\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 4.042\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.471\cos\left(6\theta\right) - 0.133\cos\left(8\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right) + 0.249\sin\left(5\theta\right)\right) \\ +R_m^{-2}\left(\begin{array}{l} 4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right) + 1.745\left(5\theta\right) \\ +0.399\sin\left(6\theta\right) \end{array} \right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) + 2.003\sin\left(6\theta\right)\right) \\ +R_m^{-4}\left(\begin{array}{l} 1.989\sin\left(4\theta\right) + 2.302\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) \\ -0.133\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right) - 0.082\sin\left(5\theta\right)\right) \\ +0.410R_m^{-2}\sin\left(5\theta\right) - 0.358R_m^{-3}\sin\left(3\theta\right) + 0.040R_m^{-4}\sin\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right) + 0.249\sin\left(5\theta\right)\right) \\ -R_m^{-2}\left(\begin{array}{l} 4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right) - 1.745\left(5\theta\right) \\ +0.399\sin\left(6\theta\right) \end{array} \right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) - 2.003\sin\left(6\theta\right)\right) \\ -R_m^{-4}\left(\begin{array}{l} 1.989\sin\left(4\theta\right) - 2.302\sin\left(5\theta\right) - 0.492\sin\left(7\theta\right) \\ -0.133\sin\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2011} = \left[\begin{array}{l} -0.037\cos\left(2\theta\right) + 0.862R_m^{-2}\cos\left(4\theta\right) \\ +R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right) - 0.008\cos\left(6\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 4.259\cos\left(2\theta\right) - 0.527\cos\left(3\theta\right) + 0.413\cos\left(5\theta\right) \\ -0.148\cos\left(6\theta\right) + 0.021\cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-4}\left(\begin{array}{l} 3.816\cos\left(2\theta\right) - 0.328\cos\left(4\theta\right) + 0.120\cos\left(5\theta\right) \\ +0.013\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{2012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.703\cos\left(3\theta\right) - 0.495\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2013} = \left[\begin{array}{l} 0.037\cos(2\theta) - 0.862R_m^{-2}\cos(4\theta) \\ +R_m^{-1}(1.790\cos(\theta) - 0.862\cos(3\theta) + 0.008\cos(6\theta)) \\ +R_m^{-3}\left(\begin{array}{l} 4.259\cos(2\theta) + 0.527\cos(3\theta) - 0.413\cos(5\theta) \\ -0.148\cos(6\theta) - 0.021\cos(7\theta) \end{array} \right) \\ -R_m^{-4}\left(\begin{array}{l} 3.816\cos(2\theta) - 0.328\cos(4\theta) - 0.120\cos(5\theta) \\ +0.013\cos(8\theta) \end{array} \right) \end{array} \right] \times 10^{-2}$$

$$d_{2111} = \left[\begin{array}{l} R_m^{-1}(3.515\sin(\theta) - 0.862\sin(3\theta) - 0.032\sin(6\theta)) \\ -R_m^{-2}(2.188\sin(2\theta) - 0.943\sin(4\theta) - 0.120\sin(6\theta)) \\ +R_m^{-3}(0.786\sin(3\theta) - 0.386\sin(4\theta) - 0.352\sin(5\theta)) \\ -0.048R_m^{-4}\sin(7\theta) \end{array} \right] \times 10^{-2}$$

$$d_{2112} = \left[\begin{array}{l} R_m^{-1}(1.406\sin(\theta) - 0.345\sin(3\theta) + 0.008\sin(5\theta)) \\ -0.039R_m^{-2}\sin(5\theta) + 0.105R_m^{-3}\sin(3\theta) \end{array} \right] \times 10^{-1}$$

$$d_{2113} = \left[\begin{array}{l} R_m^{-1}(3.515\sin(\theta) - 0.862\sin(3\theta) + 0.032\sin(6\theta)) \\ +R_m^{-2}(2.188\sin(2\theta) - 0.943\sin(4\theta) - 0.120\sin(6\theta)) \\ +R_m^{-3}(0.786\sin(3\theta) + 0.386\sin(4\theta) - 0.352\sin(5\theta)) \\ -0.048R_m^{-4}\sin(7\theta) \end{array} \right] \times 10^{-2}$$

L.6 $\phi = 180,\ 3 < R_m < 15$

$$d_{1001} = \left[\begin{array}{l} R_m^{-1}(3.515\cos(\theta) + 0.862\cos(3\theta)) \\ -R_m^{-2}(1.326\cos(2\theta) + 0.867\cos(4\theta)) \\ +R_m^{-3}(0.269\cos(3\theta) + 0.388\cos(5\theta)) - 0.258R_m^{-4}\cos(6\theta) \end{array} \right] \times 10^{-2}$$

$$d_{1002} = \left[\begin{array}{l} R_m^{-1}(1.406\cos(\theta) + 0.345\cos(3\theta)) \\ +R_m^{-3}(0.036\cos(3\theta) + 0.052\cos(5\theta)) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} R_m^{-1}(3.515\cos(\theta) + 0.862\cos(3\theta)) \\ +R_m^{-2}(1.326\cos(2\theta) + 0.867\cos(4\theta)) \\ +R_m^{-3}(0.269\cos(3\theta) + 0.388\cos(5\theta)) + 0.258R_m^{-4}\cos(6\theta) \end{array} \right] \times 10^{-2}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1012} = \left[R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.449\sin\left(3\theta\right)\right) + 0.514R_m^{-3}\sin\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.863\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) + 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\sin\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.332\cos\left(3\theta\right) - 0.511\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.487\cos\left(3\theta\right) - 3.807\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) - 3.804\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) + 0.330\sin\left(7\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.358\sin\left(3\theta\right) - 0.511\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \begin{bmatrix} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) - 8.621\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.686\sin\left(3\theta\right) - 3.804\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) - 0.330\sin\left(7\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{2011} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2012} = \begin{bmatrix} R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.703\cos\left(3\theta\right) - 0.495\cos\left(5\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2013} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.862\cos\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.527\cos\left(3\theta\right) - 0.377\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) - 0.252\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2111} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2112} = \begin{bmatrix} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.105\sin\left(3\theta\right) - 0.051\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-1}$$

$$d_{2113} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(0.786\sin\left(3\theta\right) - 0.373\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) - 0.250\sin\left(6\theta\right)\right) \end{bmatrix} \times 10^{-2}$$

L.7 $\phi = 270,\ 2 < R_m < 3$

$$d_{1001} = \left[\begin{array}{l} 0.036\cos(\theta) + 0.086R_m^{-1}\cos(3\theta) \\ -R_m^{-2}(0.219\sin(2\theta) + 0.086\sin(4\theta) - 1.282\cos(\theta)) \\ -R_m^{-3}(2.065\cos(\theta) + 0.079\cos(3\theta) + 0.035\cos(5\theta)) \\ +R_m^{-4}\left(\begin{array}{l} 0.046\sin(4\theta) + 0.024\sin(6\theta) + 1.240\cos(\theta) \\ +0.005\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^{-1}$$

$$d_{1002} = \left[\begin{array}{l} \cos(3\theta)(0.188 - (0.081\ln R_m)) + 1.406R_m^{-1}\cos(\theta) \\ +0.111R_m^{-2} - 0.049R_m^{-3}\cos(5\theta) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} 0.036\cos(\theta) + 0.086R_m^{-1}\cos(3\theta) \\ +R_m^{-2}(0.219\sin(2\theta) + 0.086\sin(4\theta) + 1.282\cos(\theta)) \\ -R_m^{-3}(2.065\cos(\theta) + 0.079\cos(3\theta) + 0.035\cos(5\theta)) \\ -R_m^{-4}\left(\begin{array}{l} 0.046\sin(4\theta) + 0.024\sin(6\theta) - 1.240\cos(\theta) \\ -0.005\cos(7\theta) \end{array} \right) \end{array} \right] \times 10^{-1}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}(1.790\sin(\theta) + 0.862\sin(3\theta) + 0.033\cos(6\theta)) \\ +R_m^{-2}(1.326\cos(2\theta) + 0.920\cos(4\theta) + 0.123\cos(6\theta)) \\ -R_m^{-3}(0.527\sin(3\theta) + 0.413\sin(5\theta) + 0.277\cos(4\theta)) \\ +R_m^{-4}(0.120\sin(5\theta) + 0.048\sin(7\theta) + 0.013\cos(8\theta)) \end{array} \right] \times 10^{-2}$$

$$d_{1012} = \left[\begin{array}{l} R_m^{-1}(7.162\sin(\theta) + 3.448\sin(3\theta)) \\ -R_m^{-3}(0.703\sin(3\theta) + 0.495\sin(5\theta)) \end{array} \right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}(1.790\sin(\theta) + 0.862\sin(3\theta) - 0.033\cos(6\theta)) \\ -R_m^{-2}(1.326\cos(2\theta) + 0.920\cos(4\theta) + 0.123\cos(6\theta)) \\ -R_m^{-3}(0.527\sin(3\theta) + 0.413\sin(5\theta) - 0.277\cos(4\theta)) \\ +R_m^{-4}(0.120\sin(5\theta) + 0.048\sin(7\theta) - 0.013\cos(8\theta)) \end{array} \right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}(0.663\cos(\theta) + 8.621\cos(3\theta)) \\ +R_m^{-2}(4.642\sin(2\theta) + 8.621\sin(4\theta) + 0.157\cos(7\theta)) \\ +R_m^{-3}\left(\begin{array}{l} 0.057\sin(8\theta) + 2.686\cos(3\theta) + 4.049\cos(5\theta) \\ -0.577\cos(7\theta) \end{array} \right) \\ -R_m^{-4}(1.989\sin(4\theta) + 2.468\sin(6\theta) - 0.803\cos(5\theta)) \end{array} \right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.358\cos\left(3\theta\right) + 0.503\cos\left(5\theta\right)\right) - 0.040R_m^{-4}\cos\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right) - 0.157\cos\left(7\theta\right)\right) \\ -R_m^{-3}\left(\begin{array}{l} 0.057\sin\left(8\theta\right) - 2.686\cos\left(3\theta\right) - 4.049\cos\left(5\theta\right) \\ +0.577\cos\left(7\theta\right) \end{array} \right) \\ +R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.468\sin\left(6\theta\right) + 0.803\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -0.173\sin\left(3\theta\right) - R_m^{-1}\left(0.663\sin\left(\theta\right) - 9.893\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.094\sin\left(3\theta\right) - 3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -4.042R_m^{-3}\sin\left(5\theta\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) \\ -2.471\cos\left(6\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.503\sin\left(5\theta\right)\right) + 0.040R_m^{-4}\sin\left(7\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2003} = \left[\begin{array}{l} -0.173\sin\left(3\theta\right) - R_m^{-1}\left(0.663\sin\left(\theta\right) - 9.893\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.094\sin\left(3\theta\right) - 3.979\cos\left(2\theta\right) - 8.621\cos\left(4\theta\right)\right) \\ -4.042R_m^{-3}\sin\left(5\theta\right) \\ +R_m^{-4}\left(\begin{array}{l} 0.773\sin\left(5\theta\right) + 0.492\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) \\ -2.471\cos\left(6\theta\right) + 0.133\cos\left(8\theta\right) \end{array} \right) \end{array} \right] \times 10^{-3}$$

$$d_{2011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.864\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right) - 0.042\sin\left(6\theta\right)\right) \\ -R_m^{-3}\left(0.208\sin\left(6\theta\right) - 0.006\sin\left(8\theta\right) + 0.379\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.060\sin\left(4\theta\right) + 0.050\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2012} = \left[R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.451\cos\left(3\theta\right)\right) + 0.510R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{2013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.864\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.862\sin\left(4\theta\right) - 0.042\sin\left(6\theta\right)\right) \\ +R_m^{-3}\left(0.208\sin\left(6\theta\right) - 0.006\sin\left(8\theta\right) - 0.379\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.060\sin\left(4\theta\right) - 0.050\cos\left(7\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{2111} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.875\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right) + 0.022\sin\left(7\theta\right)\right) \\ +0.256R_m^{-4}\cos\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

$$d_{2112} = \left[\begin{array}{l} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.036\sin\left(3\theta\right) - 0.052\sin\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$d_{2113} = \left[\begin{array}{l} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.875\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right) + 0.022\sin\left(7\theta\right)\right) \\ +0.256R_m^{-4}\cos\left(6\theta\right) \end{array} \right] \times 10^{-2}$$

L.8 $\phi = 270,\ 3 < R_m < 15$

$$d_{1001} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1002} = \left[\begin{array}{l} R_m^{-1}\left(1.406\cos\left(\theta\right) + 0.345\cos\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.105\cos\left(3\theta\right) + 0.013\cos\left(5\theta\right)\right) \end{array} \right] \times 10^{-1}$$

$$d_{1003} = \left[\begin{array}{l} R_m^{-1}\left(3.515\cos\left(\theta\right) + 0.862\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(2.188\sin\left(2\theta\right) + 0.862\sin\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.786\cos\left(3\theta\right) + 0.373\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.458\sin\left(4\theta\right) + 0.250\sin\left(6\theta\right)\right) \end{array} \right] \times 10^{-2}$$

$$d_{1011} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ -R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$d_{1012} = \left[\begin{array}{l} R_m^{-1}\left(7.162\sin\left(\theta\right) + 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.703\sin\left(3\theta\right) + 0.508\sin\left(5\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$d_{1013} = \left[\begin{array}{l} R_m^{-1}\left(1.790\sin\left(\theta\right) + 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) + 0.862\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.527\sin\left(3\theta\right) + 0.377\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.328\cos\left(4\theta\right) + 0.252\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$d_{1111} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.686\cos\left(3\theta\right) + 3.804\cos\left(5\theta\right)\right) \\ -R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) + 0.330\cos\left(7\theta\right)\right) \end{array}\right] \times 10^{-3}$$

$$d_{1112} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\cos\left(\theta\right) + 3.448\cos\left(3\theta\right)\right) \\ +R_m^{-3}\left(0.358\cos\left(3\theta\right) + 0.511\cos\left(5\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$d_{1113} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\cos\left(\theta\right) + 8.621\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(4.642\sin\left(2\theta\right) + 8.621\sin\left(4\theta\right)\right) \\ +R_m^{-3}\left(2.686\cos\left(3\theta\right) + 3.804\cos\left(5\theta\right)\right) \\ +R_m^{-4}\left(1.989\sin\left(4\theta\right) + 2.535\sin\left(6\theta\right) - 0.330\cos\left(7\theta\right)\right) \end{array}\right] \times 10^{-3}$$

$$d_{2001} = \left[\begin{array}{l} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) - 1.890\cos\left(4\theta\right) - 2.537\cos\left(6\theta\right)\right) \end{array}\right] \times 10^{-3}$$

$$d_{2002} = \left[\begin{array}{l} -R_m^{-1}\left(0.265\sin\left(\theta\right) - 3.448\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.332\sin\left(3\theta\right) + 0.511\sin\left(5\theta\right)\right) \end{array}\right] \times 10^{-2}$$

$$d_{2003} = \begin{bmatrix} -R_m^{-1}\left(0.663\sin\left(\theta\right) - 8.621\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(3.979\cos\left(2\theta\right) + 8.621\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(2.487\sin\left(3\theta\right) + 3.807\sin\left(5\theta\right)\right) \\ +R_m^{-4}\left(0.330\sin\left(7\theta\right) + 1.890\cos\left(4\theta\right) + 2.537\cos\left(6\theta\right)\right) \end{bmatrix} \times 10^{-3}$$

$$d_{2011} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ -R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) - 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2012} = \left[R_m^{-1}\left(7.162\cos\left(\theta\right) - 3.449\cos\left(3\theta\right)\right) + 0.514R_m^{-3}\cos\left(5\theta\right)\right] \times 10^{-2}$$

$$d_{2013} = \begin{bmatrix} R_m^{-1}\left(1.790\cos\left(\theta\right) - 0.863\cos\left(3\theta\right)\right) \\ +R_m^{-2}\left(0.464\sin\left(2\theta\right) - 0.858\sin\left(4\theta\right)\right) \\ +0.384R_m^{-3}\cos\left(5\theta\right) + 0.256R_m^{-4}\sin\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2111} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ +R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.867\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right)\right) + 0.258R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

$$d_{2112} = \begin{bmatrix} R_m^{-1}\left(1.406\sin\left(\theta\right) - 0.345\sin\left(3\theta\right)\right) \\ -R_m^{-3}\left(0.036\sin\left(3\theta\right) - 0.052\sin\left(5\theta\right)\right) \end{bmatrix} \times 10^{-1}$$

$$d_{2113} = \begin{bmatrix} R_m^{-1}\left(3.515\sin\left(\theta\right) - 0.862\sin\left(3\theta\right)\right) \\ -R_m^{-2}\left(1.326\cos\left(2\theta\right) - 0.867\cos\left(4\theta\right)\right) \\ -R_m^{-3}\left(0.269\sin\left(3\theta\right) - 0.388\sin\left(5\theta\right)\right) - 0.258R_m^{-4}\cos\left(6\theta\right) \end{bmatrix} \times 10^{-2}$$

APPENDIX M

Variability in Profiling - Non-interpolated LUTs

Table M.1: Timings for non-interpolated variability assessment

Run Number	Technique Timings (μs)		
	Adaptive Gauss-Legendre	Flat LUTs	Circular Arc LUTs
1	57.7	30.5	31.9
2	57.5	30.4	31.6
3	59.1	30.7	31.5
4	57.2	30.5	31.4
5	57.9	30.4	31.3
6	58.4	30.6	31.9
7	58.0	30.5	31.5
8	57.2	30.4	31.4
9	60.2	32.0	32.9
10	58.0	30.5	32.5
11	58.2	30.4	31.6
12	58.3	30.3	31.3

Continued on next page

Table M.1 – continued from previous page

Run Number	Technique Timings (μs)		
	Adaptive Gauss-Legendre	Flat LUTs	Circular Arc LUTs
13	59.6	31.7	32.9
14	57.7	30.3	31.1
15	57.3	30.2	31.1
16	57.0	30.4	31.5
17	57.5	30.5	31.7
18	56.8	30.2	31.0
19	57.6	30.5	31.2
20	57.9	31.2	31.5

Analysis of this data shows the following properties for the variability of timings within the profiling of non-interpolated LUTs.

Table M.2: Variability parameters for non-interpolated LUTs

Technique	Mean run-time (μs)	Standard Deviation
Adaptive Gauss-Legendre	58.0	0.90
Flat LUTs	30.6	0.48
Circular arc LUTs	31.6	0.55

APPENDIX N

Variability in Profiling - Interpolated LUTs

Table N.1: Timings for interpolated variability assessment

Run Number	Technique Timings (μs)		
	Adaptive Gauss-Legendre	Flat LUTs	Circular Arc LUTs
1	57.0	34.3	35.2
2	57.3	34.7	35.3
3	57.0	34.7	35.2
4	57.4	34.5	35.2
5	57.4	34.3	35.9
6	57.0	34.3	35.8
7	58.1	34.4	35.3
8	57.9	34.6	35.2
9	57.5	34.4	35.9
10	58.0	34.8	35.2
11	57.8	34.3	35.7
12	56.9	34.2	35.0

Continued on next page

Table N.1 – continued from previous page

Run Number	Technique Timings (μs)		
	Adaptive Gauss-Legendre	Flat LUTs	Circular Arc LUTs
13	59.5	35.7	36.9
14	57.3	34.4	35.8
15	57.2	34.3	35.5
16	59.4	34.3	35.3
17	57.3	34.4	35.6
18	58.7	34.6	36.4
19	58.3	34.4	35.4
20	58.7	34.4	36.7

Analysis of this data shows the following properties for the variability of timings within the profiling of interpolated LUTs.

Table N.2: Variability parameters for interpolated LUTs

Technique	Mean run-time (μs)	Standard Deviation
Adaptive Gauss-Legendre	57.8	0.78
Flat LUTs	34.5	0.33
Circular arc LUTs	35.6	0.53